UNIVERSITAT
POLITÈCNICA
DE VALÈNCIA

Continual Learning in Neural Machine Translation

April of 2025

Author: Salvador Carrión-Ponz

Director: Prof. Francisco Casacuberta

Director: Prof. Jon Ander Gómez

Acknowledgements

First and foremost, I would like to express my deepest gratitude to my PhD advisor, Prof. Francisco Casacuberta, for his unwavering support and dedication. His invaluable guidance, insightful feedback, and constant encouragement have been instrumental in the completion of this work. His highly recognized expertise in Neural Machine Translation (NMT), backed by his outstanding research track record, multiple awards, and years of teaching, has profoundly shaped my research journey.

I am equally grateful to my PhD tutor, Dr. Jon Ander Gómez, for his support and mentorship throughout my bachelor's degree, final degree project, and predoctoral research. His critical insights and constructive criticisms have greatly impacted my technical skills and the quality of this work.

A heartfelt thank you to my mother, Áurea Ponz de Tienda, whose unconditional love and support have been my foundation. Her belief in me has been a constant source of motivation throughout this endeavor.

I would also like to extend my most sincere appreciation to the rest of my family, friends, and those who are no longer with us. Their encouragement and understanding have been invaluable, especially during the challenging times of this research.

Lastly, I acknowledge all those who have contributed directly or indirectly to this thesis. Your support and contributions have been greatly appreciated.

Abstract

Neural Machine Translation (NMT) faces significant challenges in adapting to the ever-evolving nature of human language. Traditional NMT models struggle to integrate new vocabulary terms and are prone to forget previously learned information upon learning new one. This work addresses these challenges by exploring methods to enable Continual Learning (CL) in NMT, focusing on the Open-Vocabulary problem and the Catastrophic Forgetting (CF) phenomenon.

To tackle the Open-Vocabulary problem, we introduce Quasi-character-level vocabularies, which combine the flexibility of character-level models with the efficiency of higher-level encodings. This approach allows NMT models to represent any word, including unseen or rare ones, without excessively increasing sequence lengths, thereby improving model generalization and performance. Additionally, we propose incremental and continual vocabularies that leverage compositional embeddings to integrate new words seamlessly into our NMT models without the need for retraining.

In addressing the Catastrophic Forgetting problem, we investigate the influence of vocabulary domain and size on the model's retention capabilities. Next, we explore rehearsal strategies, demonstrating that minimal amounts of past data can significantly reduce the forgetting during training. Furthermore, to improve the knowledge retention without heavily relying on past data, we propose a regularization strategy that combines few-shot rehearsal with loss penalties to balance the learning of new tasks while preserving performance on previous ones.

Finally, we explore parameter-efficient adaptation methods to enable effective task-switching strategies in NMT. From this, we discover that by learning a minimal number of parameters, these methods allow NMT models to adapt to new domains, styles, or even languages without a substantial computational overhead or performance degradation on prior tasks. To complement this research, we also derive a gradient-based regularization technique for low-rank matrices that facilitates the integration of new knowledge while mitigating the CF problem.

Overall, this work advances the field of continual learning in NMT by providing practical, and thoroughly validated solutions, to the open-vocabulary problem and the CF problem, paving the way for more adaptive and efficient NMT systems capable of responding to the ever-evolving demands of natural language translation tasks.

Resumen

La traducción automática neuronal (NMT) se enfrenta a importantes retos a la hora de adaptarse a la naturaleza siempre cambiante del lenguaje natural. Los modelos tradicionales de traducción tienen dificultades para integrar nuevos términos en sus vocabularios, y son propensos a olvidar la información aprendida previamente al aprender una nueva tarea. Este trabajo aborda estos retos, explorando métodos que permitan el aprendizaje continuo en la traducción automática, centrándose tanto en el reto del vocabulario abierto como en el problema del olvido catastrófico.

Para abordar el problema del vocabulario abierto, introducimos vocabularios a nivel de cuasicarácter, que combinan la flexibilidad de los modelos a nivel de carácter con la eficacia de las codificaciones de nivel superior. Este enfoque permite a los modelos de traducción representar cualquier palabra, incluidas las no vistas o raras, sin aumentar excesivamente la longitud de las secuencias, lo que mejora la generalización y el rendimiento del modelo. Además, proponemos vocabularios incrementales y continuos que aprovechan los incrustaciones (*embeddings*) composicionales para integrar nuevas palabras en nuestros modelos de traducción sin necesidad de reentrenamiento.

Para abordar el problema del olvido catastrófico, investigamos la influencia del dominio y el tamaño del vocabulario en la capacidad de retención de nuestros modelos de traducción. Por ello, exploramos estrategias de ensayo (*rehearsal*), demostrando que con cantidades mínimas de datos anteriores pueden reducir significativamente el olvido durante el entrenamiento. Además, para mejorar la retención de conocimientos sin depender en gran medida de los datos anteriores,

proponemos una estrategia de regularización que combina el ensayo con pocos datos (*few-shot*), junto con penalizaciones en la función objetivo para equilibrar la capacidad de retención del modelo.

Por último, exploramos métodos de adaptación con parámetros eficientes para estrategias eficaces de cambio de tareas. A partir de ahí, descubrimos que, al aprender un número mínimo de parámetros, estos métodos permiten que los modelos de traducción se adapten a nuevos dominios, estilos o incluso idiomas, sin una sobrecarga computacional sustancial ni una degradación del rendimiento en tareas anteriores. Para complementar esta investigación, también derivamos una técnica de regularización basada en gradientes para matrices de bajo rango que facilita la integración de nuevos conocimientos al tiempo que mitiga el problema del olvido catastrófico.

En resumen, este trabajo supone un avance en el campo del aprendizaje continuo en la traducción automática al proporcionar soluciones prácticas, y ampliamente validadas, al problema del vocabulario abierto y al problema del olvido catastrófico, allanando el camino para sistemas de traducción automática más adaptables y eficientes, capaces de responder a las demandas, siempre en constante evolución, de las tareas de traducción de lenguaje natural.

Resum

La traducció automàtica neuronal (NMT) s'enfronta a importants reptes a l'hora d'adaptar-se a la naturalesa sempre canviant del llenguatge natural. Els models tradicionals de traducció tenen dificultats per a integrar nous termes en els seus vocabularis, i són propensos a oblidar la informació apresa prèviament en aprendre una nova tasca. Aquest treball aborda aquests reptes, explorant mètodes que permeten l'aprenentatge continu en la traducció automàtica, centrant-se tant en el repte del vocabulari obert com en el problema de l'oblit catastròfic.

Per a abordar el problema del vocabulari obert, introduïm vocabularis a nivell de quasicaràcter, que combinen la flexibilitat dels models a nivell de caràcter amb l'eficàcia de les codificacions de nivell superior. Aquest enfocament permet als models de traducció representar qualsevol paraula, incloses les no vistes o rares, sense augmentar excessivament la longitud de les seqüències, la qual cosa millora la generalització i el rendiment del model. A més, proposem vocabularis incrementals i continus que aprofiten els vectors de paraules (*embeddings*) composicionals per a integrar noves paraules en els nostres models de traducció sense necessitat de reentrenament.

Per a abordar el problema de l'oblit catastròfic, investiguem la influència del domini i la grandària del vocabulari en la capacitat de retenció dels models de traducció. Per això, explorem estratègies d'assaig (*rehearsal*), demostrant que amb quantitats mínimes de dades anteriors poden reduir significativament l'oblit durant l'entrenament. A més, per a millorar la retenció de coneixements sense dependre en gran manera de les dades anteriors, proposem una estratègia

de regularització que combina l'assaig amb poques dades (*few-shot*), juntament amb penalitzacions en la funció objectiu per a equilibrar la capacitat de retenció del model.

Finalment, explorem mètodes d'adaptació amb paràmetres eficients per a estratègies de canvi de tasques. A partir d'ací, descobrim que, en aprendre un nombre mínim de paràmetres, aquests mètodes permeten que els models de traducció s'adapten a nous dominis, estils o fins i tot idiomes, sense una sobrecàrrega computacional substancial ni una degradació del rendiment en tasques anteriors. Per a complementar aquesta investigació, també derivem una tècnica de regularització basada en gradients per a matrius de baix rang que facilita la integració de nous coneixements al mateix temps que mitiga el problema de l'oblit catastròfic.

En resum, aquest treball suposa un avanç en el camp de l'aprenentatge continu en la traducció automàtica en proporcionar solucions pràctiques, i àmpliament validades, al problema del vocabulari obert i al problema de l'oblit catastròfic, aplanant el camí per a sistemes de traducció automàtica més adaptables i eficients, capaces de respondre a les demandes, sempre en constant evolució, de les tasques de traducció de llenguatge natural.

Preface

The journey of writing this thesis has been both challenging and rewarding, marked by a profound exploration into the realms of Neural Machine Translation (NMT) and Continual Learning (CL). The rapid advancements in Artificial Intelligence (AI) have revolutionized the way machines understand and generate human language, yet the dynamic and ever-evolving nature of human language presents persistent challenges. This thesis emerges from the necessity to bridge the gap between static learning paradigms and the ever-changing linguistic landscape.

My interest in NMT was initially awakened by the impressive ability of neural networks to encode a sentence's meaning into a given latent space and accurately decode it into another language. However, as I investigated further, I became acutely aware of the limitations posed by static vocabularies and the Catastrophic Forgetting (CF) problem. The open-vocabulary problem presents a significant challenge for traditional NMT systems, as Out-of-Vocabulary (OOV) words emerge continually, complicating the translation task. Simultaneously, the CF problem prevents existing NMT models from retaining previously learned information when we adapt them to new language scenarios.

These obstacles extend beyond theoretical concerns and touch upon fundamental questions about how artificial systems can mimic the human ability to learn continually and adaptively. Inspired by this, practical solutions were investigated to enable NMT models to handle an ever-expanding vocabulary and mitigate the forgetting problem, thus moving towards truly lifelong learning NMT systems.

To provide a comprehensive understanding of this research, the thesis is organized as follows:

Chapters 1, **2**, and **3** define the context and theoretical foundations of this work. **Chapter 1**, ***Introduction***, states the problem, motivation, background, and main contributions. **Chapter 2**, ***Neural Machine Translation***, offers a concise overview of the NMT paradigm, covering neural architectures, decoding strategies, vocabulary representation, training techniques, and evaluation methods. **Chapter 3**, ***Continual Learning***, defines the CL problem, its challenges, and discusses current approaches to mitigate them.

Chapter 4, ***Experimental Framework***, discusses the experimental setup used in this work, including data sources, pre- and post-processing techniques, modeling approaches, implementation details, and evaluation metrics.

Chapters 5 and **6** represent the core of this research. **Chapter 5**, ***Addressing the Open-Vocabulary Problem in NMT***, investigates various vocabulary encoding strategies and proposes novel techniques to allow NMT systems to continually integrate new words without the need for retraining. **Chapter 6**, ***Mitigating CF in NMT***, examines methods to retain previously learned knowledge while integrating new information, along with novel parameter-efficient task-switching strategies to avoid the CF problem entirely.

The thesis concludes with **Chapter 7**, ***Conclusions***, which summarizes the key findings and discusses potential directions for future research. This is complemented by a final **Section 7.6**, ***The Fast Pace of AI Research***, which reflects on the rapid progress of AI in recent years and contextualizes the contributions of this work within that evolving landscape.

The **Appendix A**, ***Supporting Frameworks***, presents additional tools and frameworks developed during this research, offering insights into their design and practical contributions to the study. **Appendix B**, ***List of Publications***, provides a compilation of the publications authored over the course of this PhD, organized both thematically and chronologically.

Lastly, the **Bibliography** lists all references cited throughout the thesis.

The findings presented in this work represent a step toward building lifelong NMT systems that can adapt and evolve over time without forgetting prior knowledge. This thesis reflects a commitment to advancing the field of AI by tackling core challenges at the intersection of learning, memory, and language.

Notation

This section provides a concise reference describing the notation used throughout the body of the document.

Sequences and Indices

$\mathbf{x}$	Source sentence as a sequence of tokens, $\mathbf{x} = (x_1, x_2, \ldots, x_{N_x})$
$\mathbf{y}$	Target sentence as a sequence of tokens, $\mathbf{y} = (y_1, y_2, \ldots, y_{N_y})$
$\hat{\mathbf{y}}$	Predicted target sentence, $\hat{\mathbf{y}} = (\hat{y}_1, \hat{y}_2, \ldots, \hat{y}_{N_{\hat{y}}})$
N_x	Number of tokens in the source sentence
N_y	Number of tokens in the target sentence
$N_{\hat{y}}$	Number of tokens in the predicted sentence
M	Number of source-target pairs in the dataset
K	Number of tasks to learn
$\mathbf{x}^{(i)}, \mathbf{y}^{(i)}$	i-th source-target pair in the dataset
x_t, y_t	t-th token in the source and target sentences, respectively
w_j, c_j	j-th word and character in the sentence, respectively
$s_l^{(j)}$	l-th subword in the j-th word of the sentence

k_j	Total number of subwords in the j-th word
$\mathcal{T}_n$	n-th task

Neural Network Notation

θ	Model parameters
ϕ	Activation function (e.g., tanh, ReLU)
$\mathbf{W}$, $\mathbf{U}$	Weight matrices in neural networks
$\mathbf{b}$	Bias vector in neural networks
$\mathbf{A}$, $\mathbf{B}$	Low-rank matrices in Low-Rank Adaptation (LoRA)

Vocabulary and Distributions

w	Word
s	Subword
c	Char
d	Dimensionality of an embedding vector

Tasks and Datasets

$\mathcal{T}$	Task, representing a specific translation scenario (e.g., language, domain, or style)
$\mathcal{D}$	Training dataset of source-target pairs
$\mathcal{M}$	A memory abstraction

Functions

$\mathcal{L}$	Loss function (e.g., cross-entropy loss)
Ω	Regularization function (e.g., ℓ_2 regularization)
$\mathcal{R}$	Reward function (e.g., BLEU score)
$\mathbf{e}$	Embedding function
$\mathcal{C}$	Categorical distribution

$\odot$	Element-wise (Hadamard) multiplication

Probabilities

$P(\mathbf{y} \mid \mathbf{x})$	Conditional probability of a target sentence given a source sentence
$P(y_t \mid y_{<t}, \mathbf{x})$	Probability of token y_t given previous tokens $y_{<t}$ and the source sentence $\mathbf{x}$

Special Tokens

`<SOS>`	Start-of-Sentence token
`<EOS>`	End-of-Sentence token
`<UNK>`	Special token representing unknown or Out-of-Vocabulary (OOV) words
`<PAD>`	Padding token used to align sequences to the same length

Contents

Chapter 1

Introduction

This chapter introduces the problem, motivation, and challenges of integrating continual learning approaches into Neural Machine Translation (NMT) systems. It highlights the limitations of static NMT models in adapting to the dynamic nature of human language, emphasizing the need for strategies to handle the open-vocabulary problem and mitigate the Catastrophic Forgetting (CF) problem. The chapter reviews related work, identifies gaps and opportunities in current approaches, and outlines the research objectives and contributions of this work.

1.1 Background and Motivation

Machine Translation (MT) has been a key area of interest in Artificial Intelligence (AI) and Natural Language Processing (NLP) from the very beginning (Bar-Hillel, 1953; Dostert et al., 1954), aiming to automate the process of translating text and speech from one language into another. Over the years, MT has evolved significantly, transitioning through various paradigms to improve the translation quality and efficiency.

Early MT systems were rule-based, relying on extensive sets of linguistic rules and bilingual dictionaries crafted by experts (Committee, 1966; Dostert et al., 1954; Toma, 1970). While pioneering, these systems struggled with the complexity, ambiguity, and variability of natural language, leading to translations that were often literal and lacked fluency.

The advent of Statistical Machine Translation (SMT) introduced a data-driven approach, utilizing statistical models to learn translation patterns from large bilingual corpora (Koehn, 2009). SMT systems, such as phrase-based models, improved translation by capturing common phrases and their translations. However, SMT was limited by its reliance on explicit alignment and the inability to capture long-term dependencies and contextual nuances.

The introduction of Neural Machine Translation (NMT) marked a significant breakthrough. NMT models employ neural networks to model the translation process as a sequence-to-sequence problem, effectively capturing long-term dependencies and contextual information (Sutskever et al., 2014). Architectures like Recurrent Neural Networks (RNNs) (Hochreiter & Schmidhuber, 1997; Rumelhart et al., 1986), enhanced by attention mechanisms (Bahdanau et al., 2015), and later the Transformer model (Vaswani et al., 2017), have progressively advanced the State-of-the-Art (SOTA) in translation quality.

More recently, the emergence of Large Language Models (LLMs), such as GPT-3 and GPT-4, has further revolutionized the field (T. B. Brown et al., 2020; OpenAI et al., 2024). Trained on vast amounts of multilingual data, LLMs have demonstrated remarkable capabilities in understanding and generating human-like text, pushing the boundaries of what is achievable in machine translation.

Despite these advancements, NMT systems still struggle to keep pace with the ever-evolving nature of human language, which constantly introduces new vocabulary terms, expressions, and semantic shifts. This often results in outdated translations that do not reflect contemporary language use.

To address these limitations, integrating continual learning approaches into NMT systems becomes essential.

1.1.1 The Need for Continual Learning in Neural Machine Translation

Traditional NMT systems are typically trained in a static, offline manner using fixed datasets and vocabularies. Once trained, these models often remain unchanged until fine-tuned with additional data or entirely replaced. This static nature poses major challenges in real-world applications, where language is inherently dynamic and constantly evolving.

In practice, NMT systems must be able to adapt to emerging linguistic phenomena such as neologisms, slang, and domain-specific jargon to deliver accurate

and up-to-date translations. Therefore, without the ability to incorporate new knowledge on-the-fly, static NMT models quickly become outdated, compromising their usability in dynamic environments.

Continual Learning (CL) offers a promising solution to this challenge. By enabling NMT models to learn incrementally from a continuous stream of data, CL supports the development of lifelong learning systems that can adapt to new linguistic information while retaining previously acquired knowledge. This adaptability is essential for ensuring translation systems remain relevant and accurate over time.

Beyond handling linguistic evolution, CL also addresses the need for domain adaptability, as translation tasks frequently span diverse domains such as medical, legal, or technical fields, each one requiring a specialized terminology. CL facilitates incremental updates to incorporate domain-specific knowledge without the need for extensive retraining, enhancing the versatility of NMT models across different contexts.

Moreover, continual learning strategies contribute to resource efficiency because, unlike traditional approaches that require a complete retraining on large datasets, CL enables incremental updates, thus significantly reducing the computational overhead and accelerating the deployment of new model improvements.

As a result, by integrating CL strategies into NMT systems, these can evolve into robust, adaptable, and resource-efficient tools capable of meeting the demands of the ever-changing nature of human language.

1.1.2 Challenges and Limitations

Implementing a practical continual learning strategy in NMT presents several difficult challenges.

First, the open-vocabulary problem introduces a significant representation challenge, as expanding a vocabulary dynamically requires methods capable of representing new words in a way that their semantics can be incorporated into the model's word embedding space without disrupting the existing structure. Traditional approaches to incorporate new embeddings often involve retraining the model on large amounts of data, which is computationally intensive and impractical in a continual learning context.

Second, dealing with the catastrophic forgetting problem is a major challenge. As models learn new information, their performance on previously learned data can deteriorate. This occurs because updates to the model parameters for new tasks can interfere with representations learned from previous tasks, effectively overwriting important weights.

Recent advancements have proposed various methods to address some of these challenges. Rehearsal methods, where a subset of old data is presented during training on new data, help in retaining prior knowledge. Regularization-based approaches constrain the update of the model's parameters to preserve previously learned information. Dynamic architectures, including modular networks that can expand to accommodate new tasks, have shown promise in mitigating catastrophic forgetting.

Similarly, in addressing the open-vocabulary problem, approaches like low-level vocabularies (e.g., byte or character-level models) have been investigated to handle new words by modeling language at a more granular level. However, these methods often present significant efficiency limitations.

Despite these advancements, significant limitations persist. Many existing methods do not scale well with the size and complexity of SOTA NMT models. Moreover, most lack the efficiency and effectiveness to prevent catastrophic forgetting while allowing for dynamic vocabulary expansion. Consequently, developing a practical and unified solution for both the catastrophic forgetting and the open-vocabulary problem within a cohesive framework remains an open challenge.

1.2 Related Work

This section provides an overview of the progress and challenges in Neural Machine Translation (NMT) and Continual Learning (CL). It examines foundational methodologies, recent innovations, and existing limitations, laying the groundwork for the approaches proposed in this work.

1.2.1 Historical Context

The field of NMT has witnessed significant advancements since its inception. Early NMT models were primarily based on the sequence-to-sequence architecture utilizing RNNs (Sutskever et al., 2014). This architecture enabled the encoding of a source sentence into a fixed-length vector and decoding it into a target sentence. However, the limitation of fixed-length representations became an information bottleneck, especially with longer sentences.

To address this issue, Bahdanau et al., 2015 introduced the attention mechanism, allowing the model to focus on the relevant parts of the source sentence during the translation. This innovation significantly improved translation quality and became a foundational component in subsequent NMT models. The introduction of the Transformer architecture by Vaswani et al., 2017 marked a pivotal moment in NMT. By replacing recurrent layers with self-attention mechanisms, Transformers enabled parallelization and better handling of long-term dependencies, leading to substantial performance gains.

Similarly, CL emerged as a field addressing the challenge of learning sequential tasks without forgetting previously acquired knowledge. Carpenter and Grossberg, 1987 introduced the stability-plasticity dilemma, highlighting the need for neural systems to balance learning new information (plasticity) with retaining prior knowledge (stability). Subsequently, McCloskey and Cohen, 1989 formally identified the catastrophic forgetting phenomenon, where neural networks tend to forget old tasks when trained on new ones. Later on, theoretical frameworks like the Complementary Learning Systems (CLS) theory proposed by Mcclelland et al., 1995 offered insights into how biological systems might overcome forgetting, inspiring computational models that mimic such mechanisms (Blakeman & Mareschal, 2019; Pham et al., 2021).

1.2.2 Current Approaches

This subsection reviews key methods addressing the open-vocabulary problem and catastrophic forgetting phenomenon in NMT.

Open-vocabulary problem in NMT

Addressing the open-vocabulary problem is essential for an effective NMT, as languages continuously evolve over time with new words and linguistic patterns. Nowadays, subword segmentation methods have become the standard for mitigating this issue. For instance, Byte-Pair Encoding (BPE) is widely used for its ability to balance vocabulary size and coverage by iteratively merging the most frequent character pairs (Sennrich et al., 2016); Unigram Language Modeling takes a probabilistic approach assuming each subword unit in the vocabulary is independent (Kudo, 2018); and tools like SentencePiece, offer an efficient language-independent tokenization that enhances the adaptability of NMT models to different languages and scripts (Kudo & Richardson, 2018).

Character-level models have been explored to handle rare and unseen words more effectively. Costa-jussà and Fonollosa, 2016 demonstrated that character-based NMT models could alleviate the Out-of-Vocabulary (OOV) problem. However, these models often require longer training times and more computational resources due to increased sequence lengths. Lee et al., 2016 proposed fully character-level NMT without explicit segmentation, achieving competitive results but facing challenges with efficiency. Similarly, Banar et al., 2020 investigated character-level Transformers, highlighting the trade-offs between granularity and computational efficiency.

Recent advancements include token-free and byte-level models like ByT5 (Xue et al., 2021) and CANINE (Clark et al., 2021), which process raw text as a continuous stream of individual characters or bytes, eliminating the need for tokenization and handling any language or script natively. While these models address the open-vocabulary problem, they typically demand larger model sizes and more extensive training data to perform on par with subword-based models.

Mitigating the catastrophic forgetting in NMT

Mitigating catastrophic forgetting is essential for developing NMT systems capable of continual learning. Regularization-based methods constrain updates to the model's parameters to preserve important weights from previous tasks. For example, Elastic Weight Consolidation (EWC) introduced a penalty term in the loss function based on the Fisher information matrix to slow down the learning on parameters that are crucial for older tasks (Kirkpatrick et al., 2016). In contrast, Learning without Forgetting (LwF) employed a knowledge distillation technique (Hinton et al., 2015) by incorporating the outputs of the

old model into the training of the new model, thus retaining performance on prior tasks (Li & Hoiem, 2016).

Rehearsal strategies involve replaying a subset of previous data during the training on new tasks. For example, Lopez-Paz and Ranzato, 2017 alleviated the CF problem by storing a subset of the observed examples from previous tasks (episodic memory) and ensuring that the updates do not increase the loss on the stored samples. In contrast, Shin et al., 2017, instead of storing actual training data from previous tasks, trained a deep generative model that replayed previous data (synthetically) during the training of the new to prevent forgetting. In the context of NMT, Cao et al., 2021 applied rehearsal methods to continual learning, demonstrating improved retention of earlier translation capabilities.

Dynamic architectures offer another approach by expanding the model to accommodate new tasks. Progressive Neural Networks add new *columns* (networks) for each task, with lateral connections to previous columns to facilitate knowledge transfer (Rusu et al., 2016). Later, Lee et al., 2017 proposed Dynamically Expandable Networks (DEN), which grow the network structure based on the complexity of new tasks while preserving performance on previous tasks.

Parameter-Efficient Fine-Tuning (PEFT) methods aim to adapt large models to new tasks with minimal additional parameters, circumventing many of the issues from previous approaches. Houlsby et al., 2019 introduced small bottleneck layers within the Transformer architecture that could be trained for specific tasks without altering the original model parameters. Low-Rank Adaptation (LoRA) further reduced the parameter overhead by injecting trainable low-rank matrices into the existing layers to reduce the time and resources needed for fine-tuning and adapting LLMs (Hu et al., 2021).

1.2.3 Gaps and Opportunities

Despite significant progress, several challenges remain in integrating CL methods into NMT systems. Byte-, Character-, and Subword-level models, while addressing the open-vocabulary problem, often involve trade-offs between sequence length, computational efficiency, and model performance, especially on low-resource scenarios (Cherry et al., 2018; Sato et al., 2020; Sennrich & Zhang, 2019). Low-level models, such as byte and character-level models, often produce longer sequences, leading to increased training times and higher computational costs. Consequently, there is a demand for vocabulary encoding strategies that

strike a balance between the efficiency and the flexibility needed to handle new and unseen words.

Mitigating catastrophic forgetting in NMT presents complexities due to the intricacies of natural language and translation tasks. Regularization methods may struggle when new tasks differ substantially from previous ones, leading to inadequate retention of prior knowledge (Kirkpatrick et al., 2016; Li & Hoiem, 2016). Rehearsal methods, although effective, raise concerns regarding scalability and data privacy, as storing and replaying data may not always be feasible (Shin et al., 2017). Dynamic architectures can become unmanageable as they expand with each new task, posing challenges in deployment and maintenance (Han et al., 2021; Lee et al., 2017).

Accordingly, our opportunities lie in developing unified frameworks that simultaneously address the open-vocabulary problem and catastrophic forgetting phenomenon. To achieve this, exploring quasi-character-level vocabularies and compositional embeddings could offer a balance between granularity and efficiency, allowing models to incorporate new vocabulary dynamically. Additionally, integrating PEFT methods with continual learning strategies may enable effective and efficient NMT adaptations to new tasks while retaining existing learned knowledge (Hu et al., 2021; Pfeiffer et al., 2021).

Furthermore, there is a need for research focusing on the impact of linguistic diversity on continual learning in NMT. Languages with rich morphology, complex scripts, or limited resources, present unique challenges that current NMT models may not properly address (Ataman & Federico, 2018). Thus, developing language-agnostic approaches and evaluating models across a rich and diverse set of linguistic contexts could enhance the robustness and generalizability of these NMT systems (Aharoni et al., 2019; Conneau et al., 2019).

1.3 Research Objectives and Contributions

This section outlines the primary objectives and contributions of this research, focusing on addressing the critical challenges of Continual Learning in Neural Machine Translation, particularly the open-vocabulary problem and catastrophic forgetting phenomenon.

1.3.1 Objectives of the Thesis

The primary objective of this work is to develop novel strategies to enable practical Continual Learning in NMT. Specifically, this work addresses two fundamental challenges that interfere with the deployment of lifelong learning NMT systems:

1. **The Open-Vocabulary Problem:** By proposing efficient vocabulary encoding strategies that enable dynamic vocabulary expansion without the need for retraining the entire model, thereby allowing NMT systems to adapt to new and unseen words in a resource-efficient manner.

2. **The Catastrophic Forgetting Problem:** By developing methods that mitigate or circumvent the Catastrophic Forgetting problem in NMT, ensuring that they retain previously learned knowledge while incorporating new linguistic information.

By tackling these challenges, we aim to contribute towards building an NMT system that can evolve over time to operate in dynamic environments where language is constantly evolving.

1.3.2 Summary of Contributions

To address the aforementioned challenges, this work presents the following key contributions:

1. **Quasi-Character-Level Vocabularies:** We introduce a novel low-level encoding strategy that balances the granularity of character-level models with the efficiency of subword-level models (Carrión & Casacuberta, 2021, 2022c).

2. **Continual Vocabularies with Compositional Embeddings:** We propose to leverage compositional embeddings to enable dynamic vocabulary expansions on-the-fly for derived words, while a traditional approach for new semantics, to significantly reduce the need for extensive retraining when encountering OOV words (Carrión & Casacuberta, 2022a, 2023b).

3. **Strategies for Mitigating Catastrophic Forgetting:** We develop a novel approach that combines a highly effective minimal rehearsal strategy with a specific loss regularization technique to balance and mitigate the effects of catastrophic forgetting in NMT (Carrión & Casacuberta, 2022b).

4. **Task-Switching Strategies:** We explore parameter-efficient adaptations to enable interactive task-switching strategies (i.e., changing language, domain expertise, or style) to circumvent the CF problem, while using a minimal fraction of the model's parameters (Carrión & Casacuberta, 2024).

5. **Gradient-Based Regularization for Low-Rank Adapters:** We propose a gradient-based regularization approach for low-rank adapters to facilitate the integration of new knowledge into these highly compressed components (Carrión & Casacuberta, 2024).

Collectively, these contributions provide a cohesive framework for enabling practical Continual Learning in NMT, addressing both the open-vocabulary challenge and the CF problem through innovative and efficient strategies.

1.3.3 Challenges and Limitations of this Work

Despite the promising advancements presented in this work, several challenges and limitations still remain:

1. **Language Morphological Complexity:** The effectiveness of Quasi-character-level vocabularies may vary across languages with different morphological and orthographic complexities. Languages with rich morphology or complex writing systems might not benefit equally from this approach, potentially affecting the generalizability of the method.

2. **Balancing Knowledge Retention and Integration:** Achieving an optimal balance between retaining past knowledge and integrating new information within a limited parameter space is challenging, especially as the model approaches network saturation issues. Overemphasizing either aspect can lead to a suboptimal performance, thus, necessitating a careful trade-off analysis.

3. **Limitations of Low-Rank Adapters:** Incorporating new knowledge into low-rank adapters may face difficulties due to the high level of compression required within the limited parameter space. This challenge is particularly pronounced when dealing with entirely new languages or complex tasks, where the adapter's capacity may be insufficient to capture the necessary information without degrading performance.

These limitations highlight areas for future research, emphasizing the need for more efficient strategies that can accommodate a diverse set of tasks while maintaining efficiency and effectiveness in continual learning for NMT.

1.4 Thesis Organization

This section provides an overview of the thesis structure, summarizing the content and focus of each chapter to guide the reader through the progression of this work.

1.4.1 Overview of Thesis Structure

The remainder of this thesis is organized as follows:

- **Chapter 2**: Provides a comprehensive overview of Neural Machine Translation, including architectures, decoding strategies, vocabulary representations, training techniques, evaluation metrics, and specialized topics.
- **Chapter 3**: Discusses the fundamentals of Continual Learning, covering definitions, challenges, strategies, their application in NMT, and evaluation.
- **Chapter 4**: Describes the experimental framework used throughout the study, including data sources, processing techniques, modeling approaches, implementation details, and evaluation metrics.
- **Chapter 5**: Introduces methods for addressing the open-vocabulary problem in NMT, including the development of efficient low-level vocabularies and continual vocabularies.
- **Chapter 6**: Presents strategies to mitigate the catastrophic forgetting in NMT, including vocabulary influences, rehearsal and regularization strategies, and parameter-efficient methods to circumvent this problem through task-switching strategies.
- **Chapter 7**: Concludes the thesis by summarizing the contributions, discussing limitations, future work, and reflecting on their relevance in the context of the fast pace of AI research.

This structure allows for a logical progression from foundational concepts to the novel contributions of this work, offering a comprehensive understanding of Continual Learning problem in NMT.

Chapter 2

Neural Machine Translation

This chapter provides a comprehensive overview of Neural Machine Translation (NMT), including foundational architectures (e.g., RNNs, LSTMs, CNNs, Transformers, and LLMs). It discusses various decoding strategies and their relationship to the fundamental translation probability model, as well as vocabulary representations and tokenization methods to address the open-vocabulary problem. It also reviews training approaches, including offline, online, and transfer learning techniques, and examines evaluation metrics, both automatic and human-based. Finally, it explores specialized areas such as Interactive Machine Translation (IMT) and Low-Resource Machine Translation (LRMT).

2.1 Neural Architectures

Neural Machine Translation (NMT) aims to model the conditional probability of a target sentence $\mathbf{y} = (y_1, y_2, \ldots, y_{N_y})$ given a source sentence $\mathbf{x} = (x_1, x_2, \ldots, x_{N_x})$. Here, x_i and y_j represent tokens (e.g., words or subwords) in the source and target languages, while N_x and N_y indicate their corresponding lengths. Therefore, the fundamental equation of NMT can be defined as follows:

$$\hat{\mathbf{y}} = \arg\max_{\mathbf{y}} P(\mathbf{y} \mid \mathbf{x}) \tag{2.1}$$

where $\mathbf{x}$ and $\mathbf{y}$ denote the source and target sentences, and $\hat{\mathbf{y}}$ is the predicted target sentence that maximizes the conditional probability distribution.

This probability can be further decomposed using the chain rule:

$$P(\mathbf{y} \mid \mathbf{x}) = \prod_{t=1}^{N_y} P(y_t \mid y_{<t}, \mathbf{x}) \tag{2.2}$$

where $y_{<t} = (y_1, \ldots, y_{t-1})$ refers to all the previously generated tokens in the target sentence.

Consequently, modern NMT models aim to approximate $P(y_t \mid y_{<t}, \mathbf{x})$ by first encoding the source sequence $\mathbf{x}$ into a continuous representation and then decoding this representation to generate the target sequence $\mathbf{y}$, one token at a time.

To address this challenge practically, various neural architectures have been developed. These architectures differ in how they encode and decode sequences, with specific designs to capture long-term dependencies, model global context, or optimize computational efficiency. The subsections below provide an overview of the most prominent architectures employed in NMT, each illustrated with a figure to enhance understanding of their structure and functionality.

2.1.1 Recurrent Neural Networks

Recurrent Neural Networks (RNNs) are a foundational architecture in sequence modeling, designed to handle sequential data by maintaining a hidden state that captures information from previous time steps (Rumelhart et al., 1986). In NMT, RNNs process input sentences word by word, updating the hidden state $\mathbf{h}_t$ at each time step t based on the current input $\mathbf{x}_t$ and the previous hidden state $\mathbf{h}_{t-1}$:

$$\mathbf{h}_t = \phi(\mathbf{W}_h \mathbf{x}_t + \mathbf{U}_h \mathbf{h}_{t-1} + \mathbf{b}_h) \tag{2.3}$$

where ϕ is an activation function (e.g., tanh), $\mathbf{W}_h$ and $\mathbf{U}_h$ are weight matrices, and $\mathbf{b}_h$ is a bias vector.

Despite their capability to model sequences of variable length, standard RNNs suffer from vanishing and exploding gradient problems during training, which hinder their ability to capture long-term dependencies in sequences.

2.1.2 Bidirectional Recurrent Neural Networks

Bidirectional Recurrent Neural Networks (BRNNs) extend RNNs by processing the input sequence in both forward and backward directions, thereby capturing context from preceding and succeeding tokens (Sundermeyer et al., 2014):

$$\overrightarrow{\mathbf{h}}_t = \phi(\mathbf{W}_h \mathbf{x}_t + \mathbf{U}_h \overrightarrow{\mathbf{h}}_{t-1} + \mathbf{b}_h) \tag{2.4}$$

$$\overleftarrow{\mathbf{h}}_t = \phi(\mathbf{W}_h \mathbf{x}_t + \mathbf{U}_h \overleftarrow{\mathbf{h}}_{t+1} + \mathbf{b}_h) \tag{2.5}$$

$$\mathbf{h}_t = [\overrightarrow{\mathbf{h}}_t; \overleftarrow{\mathbf{h}}_t] \tag{2.6}$$

where $\overrightarrow{\mathbf{h}}_t$ represents the hidden state capturing information in the forward direction, $\overleftarrow{\mathbf{h}}_t$ represents the hidden state capturing information in the backward direction, and $[\cdot;\cdot]$ denotes concatenation, and the other terms are defined as in the RNNs equations above.

BRNNs provide a richer representation by considering both preceding and succeeding words, which is beneficial for translation tasks that require context from the entire sentence.

2.1.3 Long Short-Term Memory Networks

Long Short-Term Memory networks (LSTMs) were introduced to address the limitations of standard RNNs in capturing long-term dependencies (Hochreiter & Schmidhuber, 1997). LSTMs incorporate memory cells and gating mechanisms to regulate the flow of information:

$$\mathbf{i}_t = \sigma(\mathbf{W}_i \mathbf{x}_t + \mathbf{U}_i \mathbf{h}_{t-1} + \mathbf{b}_i) \tag{2.7}$$

$$\mathbf{f}_t = \sigma(\mathbf{W}_f \mathbf{x}_t + \mathbf{U}_f \mathbf{h}_{t-1} + \mathbf{b}_f) \tag{2.8}$$

$$\mathbf{o}_t = \sigma(\mathbf{W}_o \mathbf{x}_t + \mathbf{U}_o \mathbf{h}_{t-1} + \mathbf{b}_o) \tag{2.9}$$

$$\mathbf{g}_t = \tanh(\mathbf{W}_g \mathbf{x}_t + \mathbf{U}_g \mathbf{h}_{t-1} + \mathbf{b}_g) \tag{2.10}$$

$$\mathbf{c}_t = \mathbf{f}_t \odot \mathbf{c}_{t-1} + \mathbf{i}_t \odot \mathbf{g}_t \tag{2.11}$$

$$\mathbf{h}_t = \mathbf{o}_t \odot \tanh(\mathbf{c}_t) \tag{2.12}$$

where σ denotes the sigmoid function and $\odot$ represents element-wise multiplication. The gates $\mathbf{i}_t$, $\mathbf{f}_t$, and $\mathbf{o}_t$ control the input, forget, and output operations, respectively, enabling the network to retain relevant information over extended periods.

2.1.4 Convolutional Neural Networks

Convolutional Neural Networks (CNNs), while traditionally used in computer vision, have been adapted for NMT to model local dependencies in sequences (Gehring et al., 2017). CNN-based NMT models apply convolutional filters over the input embeddings to capture n-gram features:

$$\mathbf{h}_t = \phi\left(\sum_{k=0}^{K} \mathbf{W}_k \mathbf{x}_{t+k} + \mathbf{b}\right) \tag{2.13}$$

where K is the kernel size[1], $\mathbf{W}_k$ are filter weights, and $\mathbf{b}$ is a bias.

CNNs offer the advantage of parallel computation and fixed computational steps regardless of sequence length, making them efficient for training and inference.

2.1.5 Transformer

Transformer models have revolutionized NMT by eliminating recurrence and convolution in favor of a purely attention-based mechanism (Vaswani et al., 2017).

At the core, these models represent a sequence of input tokens (e.g., subwords) as a set of continuous embeddings, $(\mathbf{x}_1, \mathbf{x}_2, \ldots, \mathbf{x}_{N_x})$, where N_x is the length of the source sentence. Then, to capture the order of tokens, positional encodings are added to these embeddings. And finally, the model can then processes all tokens in parallel, rather than step-by-step as in RNNs, allowing it to handle long-term dependencies efficiently.

The fundamental building block of a Transformer is the self-attention mechanism, which determines how each token in the sequence relates to every other token. This is accomplished by projecting each embedding $\mathbf{x}_i$ into three vector representations: a *Query* vector $\mathbf{q}_i$, a *Key* vector $\mathbf{k}_i$, and a *Value* vector $\mathbf{v}_i$. Specifically, these projections are computed using learned weight matrices $\mathbf{W}^Q$, $\mathbf{W}^K$, and $\mathbf{W}^V$, such that:

$$\mathbf{q}_i = \mathbf{x}_i \mathbf{W}^Q, \quad \mathbf{k}_i = \mathbf{x}_i \mathbf{W}^K, \quad \mathbf{v}_i = \mathbf{x}_i \mathbf{W}^V$$

[1]Here, K refers to the kernel size of the convolutional filter, which differs from the use of K in earlier sections due to historical conventions in neural network literature.

where $\mathbf{W}^Q \in \mathbb{R}^{d \times d_k}$, $\mathbf{W}^K \in \mathbb{R}^{d \times d_k}$, and $\mathbf{W}^V \in \mathbb{R}^{d \times d_v}$ are the parameter matrices, d is the dimensionality of the input embeddings, and d_k and d_v are the dimensions of the key and value vectors, respectively.

Then, by stacking these vectors for all tokens we end up with three matrices, $\mathbf{Q}$, $\mathbf{K}$, and $\mathbf{V}$:[2]

$$\mathbf{Q} = [\mathbf{q}_1; \mathbf{q}_2; \ldots; \mathbf{q}_{N_x}] \quad \mathbf{K} = [\mathbf{k}_1; \mathbf{k}_2; \ldots; \mathbf{k}_{N_x}] \quad \mathbf{V} = [\mathbf{v}_1; \mathbf{v}_2; \ldots; \mathbf{v}_{N_x}]$$

Now, the self-attention mechanism computes a weighted combination of the value vectors, where the weights are determined by the compatibility between queries and keys:

$$\text{Attention}(\mathbf{Q}, \mathbf{K}, \mathbf{V}) = \text{softmax}\left(\frac{\mathbf{Q}\mathbf{K}^\top}{\sqrt{d_k}}\right)\mathbf{V} \tag{2.14}$$

where $\mathbf{Q}$, $\mathbf{K}$, and $\mathbf{V}$ are the matrices derived from the input embeddings. d_k is the dimensionality of the key vectors, and the softmax function ensures that attention weights sum to 1.0, to effectively distribute the attention (focus) over all other tokens in the sequence.

In the Transformer's encoder, each layer's self-attention enables every token to directly attend to every other token, capturing both local and global dependencies without having to rely on sequential processing. This allows the model to understand contextual relationships more easily. For example, a word at the beginning of the sentence can easily *attend* to another word at the end, capturing its contextual representation.

Similarly, the concept of *multi-head attention* extends this idea by using multiple sets of $(\mathbf{Q}, \mathbf{K}, \mathbf{V})$ projections (or *heads*), where each head learns to attend (focus) on different types of relationships, such as syntactic dependencies or semantic associations. Then, the outputs of all heads are concatenated and combined to form an enriched representation of each token. As a result, by stacking several layers of multi-head self-attention and feed-forward networks we achieved robust sequence encoders and decoders.

On a high level, during translation, the Transformer encoder processes the source sentence into context-rich representations. Then, the Transformer decoder uses these attention mechanisms to generate the target sentence, one token at a

[2] In this context, K refers to the key matrix used in the attention mechanism, which differs from the use of K in earlier sections due to naming conventions in neural network literature.

time, attending to both the previously generated target tokens (in the decoder's self-attention) and the full source representation (through the cross-attention layers that treat the encoder outputs as keys and values). This process allows us to model the probability distribution over the next target token at each decoding step very efficiently, ultimately producing fluent and contextually appropriate translations.

By enabling parallel computation over all token positions at the same time but, focusing on different parts of the input sequence through the multi-head attention mechanisms, Transformers have shown excellent results at capturing global dependencies to the point of becoming the standard architecture for State-of-the-Art NMT systems (Barrault et al., 2020; Tay et al., 2020).

2.1.6 Large Language Models

Large Language Models (LLMs) have emerged as a transformative paradigm in Natural Language Processing (NLP), including NMT. Unlike traditional architectures trained from scratch for specific language pairs, LLMs such as GPT (T. B. Brown et al., 2020) or PaLM (Chowdhery et al., 2022) are pre-trained on vast amounts of text data to model the probability of token sequences.

Typically, they use training objectives, such as Causal Language Modeling (CLM) (Radford et al., 2019),

$$P(\mathbf{x}) = \prod_{t=1}^{N_x} P(x_t \mid x_{<t}) \tag{2.15}$$

where $\mathbf{x} = (x_1, x_2, \ldots, x_{N_x})$ represents a sequence of tokens[3], or Masked Language Modeling (MLM):

$$P(\mathbf{x}_\mu \mid \mathbf{x}_\gamma) \tag{2.16}$$

where $\mathbf{x}_\mu$ are masked tokens predicted from their surrounding context $\mathbf{x}_\gamma$, allowing LLMs to capture both sequential and bidirectional dependencies, and enabling them to learn rich, general-purpose linguistic representations.

[3] $x_{<t}$ refers to all the previously generated tokens in the sentence

In the context of NMT, LLMs estimate the conditional probability of a target sequence $\mathbf{y} = (y_1, \ldots, y_{N_y})$ given a source sequence $\mathbf{x} = (x_1, \ldots, x_{N_x})$, which at the same time, this process is often guided by prompts or refined through task-specific fine-tuning. Fundamentally, these models simply process sequences of tokens, where the right part of conditional probability $P(y_t \mid y_{<t})$ includes both the prompt and the source sentence $\mathbf{x}$, while the tokens $y_{>t}$ represent the subsequent elements of the target sequence. For clarity, we adopt a simplified notation:

$$P(\mathbf{y} \mid \mathbf{x}) = \prod_{t=1}^{N_y} P(y_t \mid y_{<t}, \mathbf{x}) \tag{2.17}$$

This formulation highlights that each token y_t in the target sequence is predicted based on all preceding tokens $y_{<t}$ and the entire source sequence $\mathbf{x}$. Consequently, this ability to generalize across languages allows them to perform zero-shot translation by conditioning on prompts (e.g., *"Translate from English to Spanish"*):

$$P(\mathbf{y} \mid \mathbf{x}, \text{prompt}) \tag{2.18}$$

For few-shot translation, LLMs can extend this approach by incorporating examples directly into the prompt:

$$P(\mathbf{y} \mid \mathbf{x}, \text{examples}) \tag{2.19}$$

The scalability of LLMs, combined with their shared representations across languages, enables them to transfer knowledge contextually, or In-context learning (ICL). This leads to translations that are fluent, nuanced, and robust, often requiring minimal task-specific data.

2.2 Decoding Strategies

All decoding strategies in NMT aim to approximate the solution to the fundamental equation of translation described above (see Section 2.1):

$$P(\mathbf{y} \mid \mathbf{x}) = \prod_{t=1}^{N_y} P(y_t \mid y_{<t}, \mathbf{x}) \tag{2.20}$$

where $y_{<t} = (y_1, \ldots, y_{t-1})$ represents all previously generated target tokens, and the rest of the terms are defined as in the equations from the previous section (see Section 2.1).

This implies that, during the decoding phase (i.e., inference), the model provides a probability distribution over the next token y_t given all the previously generated tokens and the whole source input. Therefore, the goal of any decoding strategy is to select tokens, one at a time, to form a coherent and accurate final translation $\hat{\mathbf{y}}$.

However, since exhaustively searching over all possible sequences $\mathbf{y}$ to find the exact global maximum is computationally infeasible due to the vast size of the target space. Each strategy attempts to approximate this search problem by finding a sequence $\hat{\mathbf{y}}$ that maximizes the conditional probability distribution given by the model.

In the following subsections, we describe commonly used strategies to the fundamental translation equation.

2.2.1 Greedy Search

Greedy search is the simplest and most straightforward decoding method (Black, 2005), where, at each time step t, it selects the token with the highest conditional probability:

$$\hat{y}_t = \arg\max_{y_t} P(y_t \mid y_{<t}, \mathbf{x}) \tag{2.21}$$

While computationally efficient, it considers only the locally optimal choice at each step. This can lead to suboptimal translations since it does not take into account how current decisions might affect future word choices.

2.2.2 Beam Search

Beam search tries to provide a more balanced approach to the sequence decoding problem, by maintaining a fixed number of candidate solutions (*beams*) at each time step. That is, instead of selecting a single token per step, the algorithm stores the B most promising partial hypotheses and expands them in parallel:

$$\text{Score}\big(y_{1:t}\big) \;=\; \log P\big(y_{1:t} \mid x\big) \tag{2.22}$$

where $\text{Score}(y_{1:t})$ denotes the log-probability of a partial sequence $y_{1:t}$ given the input x (as estimated by the model), $y_{1:t}$ represents a partial translation up to step t, x is the input sequence, and $\log P$ is the log-likelihood provided by the model.

Consequently, at each step, the algorithm evaluates all possible next-token predictions $y_t \in V$, where V is the vocabulary, and selects the top B predictions with the highest scores. This process continues until a predefined termination condition is met, such as reaching an end-of-sentence token (`<EOS>`) or a maximum sequence length.

Therefore, by exploring multiple candidate solutions while controlling computational complexity, beam search often achieves better translations than greedy decoding (single-hypothesis) while remaining more practical than an exhaustive search.

2.2.3 Sampling Methods

Sampling-based decoding strategies introduce stochasticity into the inference process. This means that, instead of always picking the most probable token, these methods sample tokens from the model's output distribution.

This strategy can increase the diversity and potentially uncover more creative and varied translations. However, it may also lead to less predictable results if the probability distribution is diffuse or if there are many equally plausible translations.

In the subsections below, we describe several popular sampling-based methods:

Random sampling

Random sampling selects the next word based on the probability distribution over the vocabulary:

$$y_t \sim P(y_t \mid y_{<t}, \mathbf{x}) \tag{2.23}$$

where $\sim$ denotes sampling from a categorical distribution defined by the probabilities $P(y_t \mid y_{<t}, \mathbf{x})$.

This approach can produce highly diverse outputs but may generate less coherent or fluent translations, as it can select low-probability tokens.

Top-k sampling

Top-k sampling restricts the candidate words to the top k most probable tokens at each step and then samples from this reduced set (Radford et al., 2019). To formalize this, let V_k represent this set of top-k tokens:

$$V_k = \text{Top-k}\left(P(y_t \mid y_{<t}, \mathbf{x})\right) \tag{2.24}$$

where Top-k$(\cdot)$ is a function that returns the subset of the vocabulary V containing the k highest-probability tokens.

Then, the next token y_t is sampled from a categorical distribution $\mathcal{C}$, defined over V_k:

$$y_t \sim \mathcal{C}(V_k) \tag{2.25}$$

By limiting the sampling to a restricted set of high-probability tokens, top-k sampling allows the model to generate more coherent translations compared to unrestricted random sampling, while still maintaining some level of variability.

Nucleus sampling

Nucleus (top-p) sampling dynamically determines the candidate pool based on a probability threshold p (Holtzman et al., 2019). That is, it selects the smallest set of tokens V_p whose cumulative probability sum meets or exceeds p:

$$V_p = \{y_t \mid \sum_{y \in V} P(y \mid y_{<t}, \mathbf{x}) \geq p\} \tag{2.26}$$

The model then samples from this subset V_p, and unlike top-k sampling, where the number of candidates is fixed, nucleus sampling adapts the number of candidate tokens according to the probability distribution's shape.

This strategy can provide a better balance between diversity and quality, as it focuses on a probability mass rather than on a fixed number of tokens.

2.3 Vocabulary Representation and Tokenization

Vocabulary representation is a critical aspect in NMT for addressing the open-vocabulary problem, which arises due to the inherent diversity and dynamic nature of human languages. This section explores techniques designed to mitigate this problem, reduce computational complexity, and enhance the model's adaptability in a variety of linguistic contexts.

2.3.1 The Open-Vocabulary Problem in NMT

The Open-Vocabulary problem in NMT refers to the challenge of handling words not present in the model's predefined vocabulary, often referred to as Out-of-Vocabulary (OOV) words (Koehn & Knowles, 2017; T. Luong et al., 2014; Sennrich et al., 2016).

This issue arises because traditional NMT models rely on a fixed-size vocabulary $\mathcal{V}$ typically derived from the distinct tokens in a training set $\mathcal{D}$. More formally defined as:

$$\mathcal{D} = \left\{(\mathbf{x}^{(1)}, \mathbf{y}^{(1)}), (\mathbf{x}^{(2)}, \mathbf{y}^{(2)}), \ldots, (\mathbf{x}^{(M)}, \mathbf{y}^{(M)})\right\} \tag{2.27}$$

where M represents the total number of source-target pairs, each one consisting of a source sentence $\mathbf{x}^{(i)} = (x_1^{(i)}, \ldots, x_{Nx}^{(i)})$ and its corresponding target sentence $\mathbf{y}^{(i)} = (y_1^{(i)}, \ldots, y_{Ny}^{(i)})$. Then we define the vocabulary, $\mathcal{V}$, in this case a shared one, as the set of all distinct tokens that appear in any source or target sentence of the training set $\mathcal{D}$:

$$\mathcal{V} = \bigcup_{i=1}^{M} \Big(\{x_1^{(i)}, \ldots, x_{Nx}^{(i)}\} \cup \{y_1^{(i)}, \ldots, y_{Ny}^{(i)}\} \Big) \tag{2.28}$$

As a result, any word $w' \notin V$ will be considered an OOV, and therefore, it will be typically mapped to a special `<UNK>` token. This mapping results in the loss of specific word information, which can degrade the model's ability to compute accurate probabilities and lead to translation inaccuracies.

More precisely, let $\mathbf{x} = (x_1, \ldots, x_{N_x})$ be a source sentence and $\mathbf{y} = (y_1, \ldots, y_{N_y})$ a target sentence. Then, the model defines a conditional probability $P(\mathbf{y}|\mathbf{x})$ over sentences, but due to the presence of OOV words in either $\mathbf{x}$ or $\mathbf{y}$, there will be information gaps that will disrupt this computation.

Therefore, addressing the open-vocabulary problem is essential to develop robust vocabulary representations that minimize these information gaps, thus ensuring a comprehensive language coverage and improved translation performance.

Tokenization methods

Tokenization is a fundamental pre-processing step in NMT, where text is segmented into smaller units for efficient representation and processing. This section explores different tokenization strategies, each one tailored to model some aspect of the open-vocabulary problem and balance the model complexity with language coverage.

Word-level tokenization

Word-level tokenization treats each word as a distinct token. Formally, let $\mathcal{V}_{\text{words}}$ be the set of all word-level tokens derived from our training set $\mathcal{D}$. Therefore, a source sentence $\mathbf{x}$ can then be represented as a sequence of word tokens:

$$\mathbf{x}_{\text{words}} = \big(w_1, w_2, \ldots, w_{N_x}\big), \quad \text{where } w_j \in \mathcal{V}_{\text{words}} \tag{2.29}$$

Here, j denotes the position of the word within the sentence $\mathbf{x}$.

This tokenization, while conceptually simple, often leads to very large vocabularies, increasing model complexity and memory requirements. Moreover, any word not present in $\mathcal{V}_{\text{words}}$ will be considered as an OOV, limiting the model's ability to translate rare or unseen terms accurately.

Subword tokenization

Subword tokenization methods, such as Byte-Pair Encoding (BPE) (Sennrich et al., 2016), Unigram Language Model (Kudo, 2018), and SentencePiece[4] (Kudo & Richardson, 2018), break words into smaller units called subwords. This approach effectively balances vocabulary size and coverage, enabling the model to handle rare and OOV words by combining subword units.

In essence, these algorithms decompose words into sequences of subword units by iteratively merging the most frequent character or subword pairs in the

[4]SentencePiece is technically an efficient implementation that supports BPE, Unigram, and similar algorithms, rather than a distinct algorithm itself.

training corpus. For example, given a word w_j, it is represented as a sequence of subwords:

$$w_j = s_1^{(j)} s_2^{(j)} \dots s_k^{(j)} \tag{2.30}$$

where s is a subword unit from the subword vocabulary $\mathcal{V}_{\text{subwords}}$. Therefore, a source sentence $\mathbf{x}$ is then represented as follows:

$$\mathbf{x}_{\text{subwords}} = \big(s_1^{(1)}, \dots, s_{k_1}^{(1)},\ s_1^{(2)}, \dots, s_{k_2}^{(2)}, \dots, s_1^{(N_x)}, \dots, s_{k_{N_x}}^{(N_x)} \big) \tag{2.31}$$

where each word w_j (in a word-level sense) is now further segmented into subwords $\big(s_1^{(j)}, \dots, s_{k_j}^{(j)}\big)$ with $s_l^{(j)} \in \mathcal{V}_{\text{subwords}}$, where l denotes the position of the subword within the word w_j, and k_j represents the total number of subwords in w_j.

Consequently, subword tokenization reduces the overall vocabulary size and allows the model to represent and generate words unseen during training by composing the appropriate subword units.

Character-level tokenization

Character-level tokenization represents text as sequences of individual characters (Kim et al., 2015; Lee et al., 2016). Similar to the word-level method, let $\mathcal{V}_{\text{chars}}$ be the character set derived from the training data $\mathcal{D}$. Therefore, a source sentence $\mathbf{x}$ can then be represented as a sequence of char tokens:

$$\mathbf{x}_{\text{char}} = \big(c_1, c_2, \dots, c_{N_x}\big), \quad \text{where } c_j \in \mathcal{V}_{\text{chars}} \tag{2.32}$$

Here, j denotes the position of the character within the sentence $\mathbf{x}$.

This method offers broad coverage by theoretically allowing any word or sequence to be represented as a combination of known characters. However, it does not entirely resolve the open-vocabulary problem, especially in cases where specialized characters (e.g., emojis, symbols, characters from different languages, etc.) are underrepresented or absent in the training data. Additionally, this tokenization leads to longer sequences, increasing computational requirements and making it more challenging for the model to capture long-term dependencies.

Byte-level tokenization

Byte-level tokenization operates at the byte level, allowing for a language-agnostic representation that can handle any input text, including Unicode characters, without the need for explicit encoding schemes (Costa-jussà et al., 2017; Gillick et al., 2015; C. Wang et al., 2019).

Formally, this vocabulary only consists of all the possible byte values, leading to a fixed and small vocabulary size of 256 values:

$$\mathcal{V}_{\text{bytes}} = \{0\text{x}00, 0\text{x}00, \ldots 0\text{xFF}\} \tag{2.33}$$

Additionally, this approach is often used as a fallback mechanism for other encoding strategies.

Token-free approaches

Token-free approaches aim to eliminate the need for predefined tokens altogether, processing raw text as a continuous stream of individual characters or bytes (Clark et al., 2021; Xue et al., 2021).

Methods like continuous character representations and direct modeling of raw text with convolutional or recurrent layers fall into this category. These methods are still experimental and seek to overcome limitations associated with tokenization.

2.3.2 Vocabulary Size and Coverage

The size of the vocabulary directly affects the model's ability to represent various words and sentences. A larger vocabulary provides better coverage of the language but increases the model's complexity and computational demands. In contrast, a smaller vocabulary may simplify the model but at the cost of expressiveness and the ability to handle rare words.

In addition to this, there is strong evidence suggesting that there is an optimal vocabulary size for each translation task. For instance, Gowda and May, 2020 analyzed the impact of vocabulary size on NMT performance across multiple languages and data sizes, revealing that certain vocabulary sizes yield better results than others. Similarly, Xu et al., 2020 proposed a method to find the best token dictionary size without trial training.

Consequently, these studies highlight the importance of carefully selecting the vocabulary size to balance model complexity and language coverage, in order to improve the overall model performance.

2.3.3 Handling Out-of-Vocabulary Words

As discussed in previous sections, Out-of-Vocabulary (OOV) words pose a significant problem in NMT, as they can lead to information loss and degrade translation quality (T. Luong et al., 2015). As a result, two commonly used, non-exclusive approaches to handle this issue are:

1. **Subword Techniques:** Address OOV words by decomposing them into smaller, known subword units (Kudo, 2018; Sennrich et al., 2016). This allows the model to process words not seen during training by leveraging the compositionality of subwords.

2. **Backoff Strategies:** In cases where subword decomposition is insufficient, backoff strategies replace OOV words with generic placeholders (Jean et al., 2014; T. Luong & Manning, 2016), such as the special `<UNK>` token. Post-processing techniques, such as dictionary lookups (T. Luong et al., 2014) or external alignment-based methods, can then be applied to substitute `<UNK>` tokens with contextually appropriate terms or approximate translations.

Despite their effectiveness, both approaches face limitations. Subword techniques may fail to capture the full meaning of newly formed or domain-specific terms, while backoff strategies often rely on external resources that may not generalize well across languages or contexts. Neither approach inherently resolves the semantic gaps introduced by OOV words, particularly for terms whose meanings evolve outside of the training data (Koehn & Knowles, 2017).

2.3.4 Vocabulary Expansion Strategies

To adapt to new linguistic scenarios and minimize the impact of OOV words and new semantics, vocabulary expansion strategies can be applied. These strategies update the vocabulary dynamically during training or inference to incorporate new words as they appear.

One approach is adaptive tokenization, where the tokenizer is periodically retrained on new data to integrate recently observed vocabulary items. This enables the model to capture linguistic updates without needing extensive retraining.

Similarly, incremental vocabulary expansion involves introducing new word embeddings into the model's existing vocabulary over time. However, these newly added word embeddings must be carefully aligned with existing embeddings to maintain consistency within the model's vector space, preserving semantic relationships and ensuring a coherent translation.

2.4 Training Techniques

Modern NMT systems rely on a variety of training paradigms, each one tailored to address specific challenges such as limited data availability, domain adaptation, and dynamic environments. This section provides an overview of these techniques, categorized into offline learning, online learning, transfer learning, and reinforcement learning, highlighting their objectives, methodologies, and practical applications.

2.4.1 Offline Learning

Offline learning involves training NMT models on a fixed, predefined dataset, where all training data is available at the start of training (Bishop, 2006). The model parameters are optimized iteratively to minimize a predefined loss function, typically over multiple epochs.

Supervised learning

Supervised learning is the most common training approach in NMT (Barrault et al., 2020; Kocmi et al., 2023). It requires a parallel dataset $\mathcal{D} = \{(\mathbf{x}^{(i)}, \mathbf{y}^{(i)})\}_{i=1}^{M}$, where $\mathbf{x}^{(i)}$ and $\mathbf{y}^{(i)}$ represent source and target sentences, respectively.

The objective is to maximize the conditional likelihood of the target sentence given the source sentence or, equivalently, minimize the Negative Log-Likelihood (NLL) loss:

$$\mathcal{L}_{\text{NLL}} = -\sum_{i=1}^{M}\sum_{t=1}^{N_y} \log P(y_t^{(i)} \mid y_{<t}^{(i)}, \mathbf{x}^{(i)}; \theta) \tag{2.34}$$

where θ denotes the model parameters, $y_t^{(i)}$ is the t-th target token of the i-th sentence pair, $y_{<t}^{(i)}$ includes all tokens up to $N_y - 1$, M is the total number of

sentence pairs in the dataset, and N_y is the length of the target sequence for a given pair $(\mathbf{x}, \mathbf{y})$.

Unsupervised learning

Unsupervised NMT aims to leverage monolingual data to train NMT models without direct access to parallel corpora (Artetxe et al., 2018; Lample et al., 2018). Techniques such as back-translation (Sennrich et al., 2015) and dual learning (He et al., 2016) are commonly used.

- **Back-Translation:** Generates synthetic parallel data by translating monolingual target sentences into the source language using a reverse model $P(\mathbf{x}|\mathbf{y}; \theta_{\text{rev}})$. Then, the synthetic pairs $(\tilde{\mathbf{x}}, \mathbf{y})$ are then used to train the forward model $P(\mathbf{y}|\mathbf{x}; \theta)$
- **Dual Learning:** Simultaneously trains the forward translation model $P(\mathbf{y}|\mathbf{x}; \theta_{X \to Y})$ and the reverse model $P(\mathbf{x}|\mathbf{y}; \theta_{Y \to X})$, by exploiting the duality between translation directions.

Semi-supervised learning

Semi-supervised learning combines both, parallel (supervised) and monolingual (unsupervised) data to improve NMT models through a multi-part loss (Cheng et al., 2016):

$$\mathcal{L}_{\text{semi}} = \alpha \mathcal{L}_{\text{NLL}} + (1 - \alpha) \mathcal{L}_{\text{unsup}} \tag{2.35}$$

where $\mathcal{L}_{\text{NLL}}$ is the negative log-likelihood on parallel data, $\mathcal{L}_{\text{unsup}}$ represents the unsupervised loss (e.g., back-translation loss), and α is a hyperparameter balancing the two terms.

2.4.2 Online Learning

Online learning updates the model parameters incrementally as new data arrives, allowing the model to adapt to dynamic environments (Kirkpatrick et al., 2016; Littlestone & Warmuth, 1994; Peris & Casacuberta, 2019). However, a key challenge is Catastrophic Forgetting (CF), where the model forgets previously learned information when updated with new data (Mcclelland et al., 1995; Parisi et al., 2018).

Incremental learning

Incremental learning (IL) in NMT involves updating the model incrementally as new data becomes available without retraining from scratch (Liang et al., 2024; Parisi et al., 2018; Schwarz et al., 2018).

For example, suppose that after training a model on an initial dataset or task $\mathcal{T}_{old}$, the model parameters are θ^*. Then, when new dataset or task $\mathcal{T}_{new}$ arrives, the model is updated to θ by minimizing a modified objective:

$$\mathcal{L}_{\text{inc}}(\theta) = \mathcal{L}_{\text{NLL}}(\mathcal{T}_{new}; \theta) + \gamma\Omega(\theta, \theta^*) \tag{2.36}$$

where $\Omega(\theta, \theta^*)$ is a regularization term encouraging the new parameters θ to stay close to the old ones θ^*, and γ is a hyperparameter.

This approach is beneficial for adapting to new domains or incorporating recent language usage without retraining the entire model.

Continual learning

Continual Learning (CL) generalizes incremental learning to a long sequence of tasks (or data distributions) (Chen et al., 2018; Parisi et al., 2019; Parisi et al., 2018), aiming to maintain performance on previously seen languages or domains while learning new ones. Thus, this work focuses on it as a core research area.

Formally, consider a sequence of datasets or tasks $\{\mathcal{T}_1, \mathcal{T}_2, \ldots, \mathcal{T}_K\}$. After learning these different data distributions $(1, ..., K-1)$, the model faces $\mathcal{T}_K$:

$$\mathcal{L}_{\text{CL}}(\theta) = \mathcal{L}_{\text{NLL}}(\mathcal{T}_K; \theta) + \sum_{n=1}^{K-1} \lambda_n \Omega(\theta, \theta_n^*) \tag{2.37}$$

where θ_n^* are the parameters for past tasks, and λ_n balances the importance of retaining old knowledge.

This framework supports numerous approaches, including regularization-based (Kirkpatrick et al., 2016; Li & Hoiem, 2016)), replay-based (Shin et al., 2017), parameter isolation methods (Hu et al., 2021; Pfeiffer et al., 2021)), etc.

2.4.3 Transfer Learning

Transfer learning leverages knowledge from one domain or task to improve the performance on another (Pratt, 1992). In NMT, it often involves pre-training on a high-resource language pair and then fine-tuning on a low-resource one (Neubig & Hu, 2018; Zoph et al., 2016).

Fine-tuning

Fine-Tuning (FT) is a two-step process where a model is pre-trained on a large, general-purpose dataset and then adapted to a smaller, task-specific dataset (Oquab et al., 2014). This approach leverages pre-trained knowledge, such as syntactic and semantic patterns, to accelerate convergence and improve performance in specialized domains.

The objective function typically remains the same as in supervised learning:

$$\mathcal{L}_{\text{Fine-tune}} = -\sum_{i=1}^{M}\sum_{t=1}^{N_y} \log P(y_t^{(i)} \mid y_{<t}^{(i)}, \mathbf{x}^{(i)}; \theta) \tag{2.38}$$

This technique is especially effective for low-resource scenarios or specialized tasks where training a model from scratch is infeasible.

Multi-task learning

Multi-Task Learning (MTL) involves training the model on multiple related tasks simultaneously (e.g., multi-language training, auxiliary tasks,...), sharing representations and parameters across tasks, with the goal of improving generalization and efficiency (Dong et al., 2015; M.-T. Luong et al., 2015).

Formally, let $\mathcal{T}_1, \mathcal{T}_2, \ldots, \mathcal{T}_K$ represent different tasks or datasets. The objective function becomes:

$$\mathcal{L}_{\text{MTL}} = \sum_{n=1}^{K} \alpha_n \mathcal{L}_{\mathcal{T}_n} \tag{2.39}$$

where α_n is a weighting factor for task $\mathcal{T}_n$.

Parameter-Efficient Fine-Tuning

Parameter-Efficient Fine-Tuning (PEFT) methods aim to adapt large pre-trained models to new tasks or domains by updating only a small subset of parameters, thus reducing computational and storage costs (Houlsby et al., 2019; Pfeiffer et al., 2021).

One approach is to partition parameters into two sets: θ_ϕ (main model parameters, kept frozen) and θ_a (adaptation parameters). The objective remains similar to the NLL loss, but only θ_a is updated:

$$\mathcal{L}_{\text{PEFT}}(\theta_\phi, \theta_a) = -\sum_{i=1}^{M}\sum_{t=1}^{N_y} \log P(y_t^{(i)} \mid y_{<t}^{(i)}, \mathbf{x}^{(i)}; \theta_\phi, \theta_a) \tag{2.40}$$

For example, in this work we make use of Low-Rank Adaptation (LoRA) (Hu et al., 2021), which applies low-rank adaptations to the weight matrices of the model:

$$\mathbf{W} \leftarrow \mathbf{W} + \mathbf{AB} \tag{2.41}$$

where $\mathbf{W}$ represents the original weight matrix of the model, $\mathbf{A}$ and $\mathbf{B}$ are the low-rank matrices learned to add the task-specific changes. This approach reduces the number of trainable parameters and makes the process more efficient.

2.4.4 Reinforcement Learning

Reinforcement Learning (RL) can be applied to NMT to optimize metrics that are non-differentiable, such as BLEU or TER, using a reward-based framework. That is, the model acts as an agent generating translations (actions) given source sentences (states), receiving rewards based on translation quality (Nguyen et al., 2017; Ranzato et al., 2016).

Formally, the model generates a sequence $\mathbf{y}$ for an input $\mathbf{x}$ and receives a reward $R(\mathbf{y}, \mathbf{y}_{\text{ref}})$ based on the translation quality. The objective is to maximize the expected reward:

$$\mathcal{L}_{\text{RL}} = -\mathbb{E}_{\mathbf{y}\sim P(\cdot|\mathbf{x};\theta)}[\mathcal{R}(\mathbf{y}, \mathbf{y}_{\text{ref}})] \tag{2.42}$$

where $P(\cdot \mid \mathbf{x}; \theta)$ is the model's predicted distribution, and $R(\mathbf{y}, \mathbf{y}_{\text{ref}})$ is the reward function comparing $\mathbf{y}$ to the reference translation $\mathbf{y}_{\text{ref}}$.

Techniques such as REINFORCE (Williams, 1992) or actor-critic methods (Bahdanau et al., 2016) are commonly employed to approximate the gradient of the reward function and update model parameters effectively.

2.5 Evaluation Metrics

Evaluation of NMT systems commonly relies on two main categories of metrics: Automatic and Human-based.

Automatic metrics provide a fast and inexpensive way to compare system outputs against reference translations. However, these may fail to fully capture nuanced linguistic and semantic aspects of quality. In contrast, human evaluation, while more resource-intensive, offers a more direct and comprehensive assessment of translation fidelity and fluency.

2.5.1 Automatic Evaluation Metrics

Automatic evaluation metrics provide a fast, cost-effective, and consistent way to assess the quality of the translations generated by a NMT system. Specifically, these metrics evaluate how closely the system's translations (hypotheses) align with the expected target translations (references) for a given source sentence. Thus, by comparing the hypotheses to the reference translations, they aim to measure translation accuracy, fidelity, and fluency in a systematic way. However, these metrics often prioritize surface-level lexical and structural correspondences over deeper semantic nuances.

In the following subsections, we discuss the most commonly used metrics in Machine Translation.

BLEU score

The Bilingual Evaluation Understudy (BLEU) score is a precision-based metric that measures the n-gram overlap between machine-generated translations and one or more reference translations (Papineni et al., 2002). Consequently, higher scores indicate a greater lexical similarity (in terms of n-grams) between the candidate translation and the reference, and thus suggest a better alignment

with the expected target translation. Also, it is the facto standard metric in NMT (Barrault et al., 2020; Kocmi et al., 2023).

Formally, let p_n represent the modified n-gram precision for n-grams of order n, and w_n the weights assigned to each n-gram order (commonly uniform and summing to 1). Then, given a candidate translation of length c, and a reference translation of length r, the BLEU score is computed as:

$$\text{BLEU} = \text{BP} \cdot \exp\left(\sum_{n=1}^{N} w_n \log p_n\right) \tag{2.43}$$

where:

- BP is the *brevity penalty*, introduced to penalize overly short candidate translations:
$$\text{BP} = \begin{cases} 1 & \text{if } c > r \\ e^{(1-r/c)} & \text{if } c \leq r \end{cases}$$
- p_n is the modified precision of n-grams of size n, which counts how many n-grams from the candidate appear in the reference.
- w_n represents the weight assigned to each n-gram order n, typically $w_n = \frac{1}{N}$ for $n = 1$ to 4.

BLEU scores are often multiplied by 100 and reported on a scale from 0 to 100, where higher values indicate better translation performance for the evaluated system.

ChrF

Character F-score (ChrF) is a metric that computes the F_β score based on the precision (P) and recall (R) of character-level n-grams (Popović, 2015). This metric is particularly useful for morphologically rich languages where word-level metrics might miss important variations.

The formula for ChrF is:

$$\text{ChrF} = \frac{(1 + \beta^2) \cdot P \cdot R}{\beta^2 \cdot P + R} \tag{2.44}$$

where:

- P is the precision, calculated as the ratio of matched character n-grams between the candidate and the reference translation out of all character n-grams in the candidate translation.
- R is the recall, calculated as the ratio of matched character n-grams between the candidate and the reference translation out of all character n-grams in the reference translation.
- β is a weighting parameter that emphasizes recall over precision, often set to $\beta = 2$ for standard configurations.

Translation Edit Rate

Translation Edit Rate (TER) is a metric that measures the number of edit operations (insertions, deletions, substitutions, and optional shifts of word blocks) required to transform the machine-generated translation into the reference translation (Snover et al., 2006). This metric reflects how much post-editing effort a human translator needs to spend to fix errors in the candidate output.

The formula for TER is:

$$\text{TER} = \frac{E}{L} \tag{2.45}$$

where:

- E is the total number of edit operations needed to match the reference translation.
- L is the length of the reference translation.

A lower TER value indicates fewer edits, and consequently, a better alignment between the machine-generated output and the reference translation.

METEOR

Metric for Evaluation of Translation with Explicit ORdering (METEOR) is a metric that evaluates translations based on the harmonic mean of unigram precision (P) and recall (R) (Lavie & Agarwal, 2007), considering linguistic features such as synonyms, stemming, and word order alignment, to make it more robust to variations in lexical choice and syntax. This flexibility allows it to correlate more strongly with human judgment compared to simpler n-gram-based metrics.

The formula for METEOR is:

$$\text{METEOR} = \frac{(1+\beta) \cdot P \cdot R}{\beta \cdot P + R} \tag{2.46}$$

where:

- P is the unigram precision, calculated as the ratio of matched unigrams in the candidate translation out of all unigrams in the candidate translation.
- R is the unigram recall, calculated as the ratio of matched unigrams in the candidate translation out of all unigrams in the reference translation.
- β is a weighting parameter that emphasizes recall over precision, often set to $\beta = 9$ in standard configurations.

BERTScore

Bidirectional Encoder Representations from Transformers Score (BERTScore) is a metric that evaluates the translations at a semantic level by leveraging contextual embeddings from pre-trained language models[5] (T. Zhang et al., 2019). Unlike traditional surface-based metrics, this metric aligns candidate and reference translations at the token level based on their contextual representations, enabling it to capture nuanced meaning beyond lexical similarities.

The score for each word is based on the cosine similarity with its counterpart in the reference, and the overall score is the F_1 score of these similarities:

$$\text{BERTScore} = F_1 \left(\text{cosine}(\mathbf{e}_y, \mathbf{e}_{y^{\text{ref}}})\right) \tag{2.47}$$

[5]BERT (Devlin et al., 2018) is commonly used as the default pre-trained language model for BERTScore

where:

- $\mathbf{e}_y$ and $\mathbf{e}_{y^{\text{ref}}}$ are embeddings for the candidate and reference tokens, respectively
- $\text{cosine}(\cdot, \cdot)$ computes cosine similarity

Higher scores imply greater semantic alignment.

2.5.2 Human Evaluation

Human evaluation is considered the gold standard for assessing the quality of translations, as it captures linguistic nuances and contextual fidelity that automatic metrics often overlook (Bentivogli et al., 2018; Graham et al., 2013). Unlike automated approaches, human evaluation focuses on subjective judgments about translation quality, based on several criteria:

1. **Fluency:** Measures how grammatically correct and natural the translation sounds in the target language. Evaluators consider whether the output reads like a sentence written by a native speaker.
2. **Adequacy:** Assesses how accurately the meaning of the source text is preserved in the translation. This includes capturing semantic nuances and ensuring that no critical information is lost or distorted.
3. **Fidelity:** Evaluates the consistency of the translation with the stylistic and cultural context of the source text, ensuring that domain-specific terminology or conventions are respected.

Human evaluation can be conducted through different methodologies:

- **Direct Assessment (DA):** Human raters assign scores to translations on a continuous scale (e.g., 1 to 100) based on overall quality (Bentivogli et al., 2018).
- **Ranking:** Evaluators rank multiple translations of the same source sentence in order of preference (Moosa et al., 2024).
- **Error Analysis:** Raters identify and categorize specific errors in translations (e.g., grammatical, lexical, or semantic errors) (Popović, 2018).

While human evaluation provides invaluable insights, it is resource-intensive, requiring time, expertise, and consistency among evaluators. To mitigate

these challenges, standardized guidelines, such as the Multidimensional Quality Metrics (MQM) framework (Lommel et al., 2014), are often employed to ensure systematic and reproducible assessments.

Consequently, complementing automatic metrics with human evaluation allows us to gain a more accurate understanding of the translation quality, especially in contexts where the fluency and adequacy take precedence over superficial or morphological matches against the reference translations.

2.6 Specialized Topics

Certain advanced topics in NMT focus on enhancing the adaptability and performance of a MT system in very specific scenarios. This section briefly explores Interactive Machine Translation (IMT), which promotes human-AI collaboration, and Low-Resource Machine Translation (LRMT), which addresses challenges in translating languages with limited data resources.

2.6.1 Interactive Machine Translation

Interactive Machine Translation (IMT) is a collaborative approach where human translators interact with the NMT system to iteratively refine translations (Peris & Casacuberta, 2019). That is, the system suggests translations, and the human translator provides feedback by accepting, modifying, or rejecting the output.

Formally, let $\mathbf{x}$ represent the source sentence, $\mathbf{y}_{1:t-1}$ be the confirmed prefix up to time $t-1$ $(y_1, \ldots, y_t - 1)$, and y_t the next word generated by the system. Then, the IMT model predicts the next word as follows:

$$y_t = \arg\max_{y \in V} P(y \mid \mathbf{y}_{1:t-1}, \mathbf{x}; \theta) \tag{2.48}$$

where V is the vocabulary, and θ denotes the model parameters. If the translator rejects the suggestion y_t, the IMT system recalculates the probability distribution based on updated constraints, potentially applying penalties to rejected words or phrases.

This feedback loop allows the system to incorporate human corrections and adjust translations in real-time, enhancing translation accuracy and fluency.

2.6.2 Low-resource Machine Translation

Low-Resource Machine Translation (LRMT) focuses on translating language pairs with limited parallel data (Cherry et al., 2018; Zoph et al., 2016). This scarcity makes it difficult for NMT models to learn robust representations, thus leading to poor translation quality.

To overcome these limitations, researchers have developed various strategies based mainly on transfer learning and unsupervised learning methods (Haque et al., 2021; Sennrich et al., 2015).

2.7 Chapter Summary

In this chapter, we presented an extensive overview of the core components that make modern NMT systems. We began by discussing the evolution of neural architectures, from early RNNs and LSTMs to CNNs, ending with the Transformer model (Section 2.1). These architectures vary in their strategies to capture long-term dependencies and offer different trade-offs in terms of parallelization, performance, and training complexity. We also introduced LLMs, which have recently emerged as powerful, general-purpose language models that can be adapted to a wide variety of NMT tasks via prompting.

Next, we surveyed the most common decoding strategies (Section 2.2), including greedy search, beam search, and sampling methods, with which to approximate the optimal translation given the model's output probabilities. Each one, seeking to balance the trade-offs between computational efficiency and search exploration.

We then turned our attention to vocabulary representation and tokenization methods (Section 2.3), highlighting the open-vocabulary problem in NMT. Accordingly, various tokenization approaches, such as word-, subword-, character-level, and byte-level, were discussed, emphasizing their impact on vocabulary coverage, model complexity, and the handling of OOV words. Additionally, we discussed vocabulary expansion strategies designed to adapt NMT systems to the ever-evolving human language landscape.

Moving on to training techniques (Section 2.4), we explored offline, online, transfer learning, and reinforcement learning approaches. These training paradigms address a wide range of problems, from limited data availability and domain adaptation to learning from non-differentiable reward signals. Special attention was paid to incremental learning and continual learning methods, which incre-

mentally update model parameters to accommodate new data, emphasizing challenges such as Catastrophic Forgetting.

In terms of evaluation (Section 2.5), we studied both automatic metrics (e.g., BLEU, ChrF, TER, METEOR, and BERTScore) and human-based evaluation strategies. Although automatic metrics are essential and widely used for large-scale comparisons, human evaluation remains the most reliable strategy for translation quality, though it is more resource-intensive.

Lastly, we covered specialized topics (Section 2.6) that extend NMT capabilities to particular challenges and use cases. This included Interactive Machine Translation, which integrates human translators in an interactive feedback loop, and Low-Resource Machine Translation, targeting scenarios where bilingual corpora are scarce.

As a result, this chapter provides the necessary conceptual and technical background to understand the complexity of current NMT systems, laying the groundwork for the subsequent chapters that study the continual learning problems and their application to NMT.

Chapter 3

Continual Learning

This chapter provides a detailed examination of the Continual Learning (CL) problem, defining its importance in Machine Learning (ML) and its relevance to Neural Machine Translation (NMT). It identifies key challenges such as catastrophic forgetting, data distribution shifts, and scalability constraints. The chapter reviews various strategies to address these challenges, including rehearsal methods, regularization-based techniques, dynamic architectures, and Complementary Learning Systems (CLS). Additionally, it examines the application of CL in NMT, discussing specific challenges, existing approaches, and the impact on translation quality. Finally, it covers the evaluation of continual learning methods in NMT, focusing on relevant metrics and benchmarks specific to translation tasks.

3.1 Definition and Importance

Continual Learning (CL), also known as *Lifelong Machine Learning*, is a paradigm in ML where models are trained to learn from a continuous stream of data or tasks, without forgetting previously acquired knowledge (Chen et al., 2018; Parisi et al., 2019). In contrast to the offline and static approaches described in the previous chapter (see Section 2.4), where models are trained on a fixed dataset and then deployed without further updates, continual learning aligns more closely with online and incremental learning paradigms. However, while online learning focuses on updating a model as new data arrives and

incremental learning addresses adding new knowledge without revisiting all past data, continual learning emphasizes retaining and integrating knowledge across a potentially large sequence of tasks. This expanded focus includes maintaining performance on earlier tasks, adapting to shifts in data distribution, and managing growth in model complexity or data volume over time.

Consequently, researching continual learning strategies is essential for developing intelligent systems that can adapt to new tasks, domains, or environments, much as described for incremental or adaptive training strategies in the previous chapter, but with a stronger emphasis on long-term retention and knowledge accumulation.

Formally, let $\mathcal{T} = \{\mathcal{T}_1, \mathcal{T}_2, \ldots, \mathcal{T}_K\}$ represent a sequence of tasks, where each $\mathcal{T}_n$ introduces new data distributions, and K the total number of tasks to learn. Then, a model parameterized by θ must learn each task $\mathcal{T}_n$ without forgetting the knowledge acquired from previous tasks $\{\mathcal{T}_1, \ldots, \mathcal{T}_{n-1}\}$. Thus, we aim to minimize a cumulative loss function:

$$\mathcal{L}_{CL}(\theta) = \sum_{n=1}^{K} \lambda_n \mathcal{L}_{\mathcal{T}_n}(\theta) \tag{3.1}$$

where $\mathcal{L}_{\mathcal{T}_n}(\theta)$ represents the loss for task $\mathcal{T}_n$, and λ_n are task-specific weights that balance the importance of new and old knowledge.

In the context of NMT, continual learning enables models to adapt to new language pairs, domains, or styles without the need for retraining from scratch, much like fine-tuning or domain adaptation techniques discussed previously. Yet, continual learning extends these concepts by ensuring that as the system evolves with new linguistic patterns, it maintains high translation quality for earlier tasks and knowledge domains.

3.2 Challenges in Continual Learning

Despite its potential benefits, continual learning presents several significant challenges that need to be addressed for effective implementation. These include issues related to catastrophic forgetting, handling data distribution shifts, and ensuring scalability under computational and memory constraints.

3.2.1 Catastrophic Forgetting

The Catastrophic Forgetting (CF) refers to the tendency of neural networks to forget previously learned information upon learning new one (Carpenter & Grossberg, 1987; McCloskey & Cohen, 1989). This phenomenon occurs because, in the process of updating the model parameters to fit new data, these updates can interfere with the representations learned from previous data.

Formally, let's imagine a model that has learned task A and is subsequently trained on task B. Similarly, let $\mathcal{L}_A(\theta)$ and $\mathcal{L}_B(\theta)$ represent the loss functions for tasks A and B, respectively. Then, when the model trained on task A is trained on task B, thus, minimizing $\mathcal{L}_B(\theta)$, it may potentially increase $\mathcal{L}_A(\theta)$, thereby degrading performance on task A. This process can be mathematically expressed as:

$$\theta^* = \arg\min_{\theta} \mathcal{L}_B(\theta) \Rightarrow \mathcal{L}_A(\theta^*) \geq \mathcal{L}_A(\theta_{\text{initial}}) \tag{3.2}$$

where θ_{initial} are the parameters after training on task A. Addressing CF requires methods that prevent or mitigate this interference, especially in sequential tasks where preserving earlier knowledge is essential.

In the context of NMT, this problem results in a significant drop in the translation quality for language pairs or domains in which the model was previously proficient.

3.2.2 Data Distribution Shifts

In real-world applications, the data distribution often changes over time in a phenomenon known as *concept drift* or *data distribution shift* (Hinder et al., 2023).

As a result, models trained under the assumption of a stationary data distribution may perform well at first, but over time, when faced with new data that differs significantly from the training data, they will perform poorly. In NMT, this often occurs due to changes in language use, the emergence of new jargon and terms, etc.

3.2.3 Scalability and Computational Constraints

A key challenge in continual learning is managing computational and memory efficiency (Chen et al., 2018; De Lange et al., 2022). While it is not mandatory to store all previously seen datasets, a naive approach might attempt to store a substantial amount of past data for replay, which leads to impractical memory requirements as tasks accumulate. Instead, continual learning strategies should cleverly limit or structure their memory usage, either through a carefully selected subset of past examples, compressed representations, adapters, generative replay, or other methods, so that the growth in storage and computation remains manageable. This strategy ensures that systems can scale efficiently to handle an ever-growing number of tasks, without incurring in exponential resource consumption.

Formally, let's consider a sequence of tasks $\{\mathcal{T}_1, \mathcal{T}_2, \ldots, \mathcal{T}_K\}$, with their corresponding datasets $\{\mathcal{D}_1, \mathcal{D}_2, \ldots, \mathcal{D}_K\}$[1]. In this context, we use $\mathcal{M}$ as a *memory abstraction* or *memory budget* that can take different forms depending on the specific approach. For example, instead of storing the entire dataset for each task, we could use a subset of this, $\mathcal{M}_n \subseteq \mathcal{D}_n$. Similarly, this memory set $\mathcal{M}_n$ could correspond to a carefully selected subset of task-specific parameters (e.g., adapters), or even a model used for generative replay, all of which require significantly less storage than the original dataset. Therefore, by ensuring that these memory sets remain within a fixed and slowly growing budget, the total memory usage $\mathcal{M}$ becomes a manageable linear function with respect to the number of tasks K, regardless of the specific approach used to store or encode the information from the past tasks:

$$\mathcal{M} = \sum_{n=1}^{K} |\mathcal{M}_n| \tag{3.3}$$

where $|\mathcal{M}_n|$ is the size of the subset stored for task $\mathcal{T}_n$, and K the total number of tasks.

Consequently, by restricting each $|\mathcal{M}_n|$ to a small, constant size or ensuring that it does not grow significantly with K, we can keep $\mathcal{M}$ bounded to grow slowly. For instance, if each $|\mathcal{M}_n|$ is capped at a constant c:

$$|\mathcal{M}_n| \leq c \quad \forall n \tag{3.4}$$

[1]We assume one dataset per task

then:

$$\mathcal{M} = \sum_{n=1}^{K} |\mathcal{M}_n| \leq \sum_{n=1}^{K} c = c \cdot K \tag{3.5}$$

Alternatively, if we maintain a global fixed budget B for memory and replace older examples as new tasks arrive, we can ensure:

$$\mathcal{M} = \sum_{n=1}^{K} |\mathcal{M}_n| \leq B \tag{3.6}$$

where B does not depend on K.

Specifically, the first case corresponds with the regularization approaches presented in Section 6.3.2, while the second case corresponds to the rehearsal strategies and parameter-efficient adaptation methods described in Sections 6.3.1 and 6.4, respectively.

Because of this, it is imperative to design and implement scalable algorithms that operate under strict resource limitations, ensuring that continual learning models remain both memory- and computation-efficient as more tasks are introduced.

3.3 Strategies for Continual Learning

To address these challenges, several strategies have been proposed, including rehearsal methods, regularization techniques, and dynamic architectures, among many others. This section introduces some of these strategies, explaining their underlying mechanisms and their relevance to the Continual Learning problem.

3.3.1 Rehearsal Methods

Rehearsal methods try to mitigate the catastrophic forgetting by replaying past data during the training on new tasks (Robins, 1995). This often involves storing a subset of past data (experience replay), or more recently, generating pseudo-data representing past tasks through the use of generative models (pseudo-rehearsal) (Qu et al., 2024; Shin et al., 2017).

Formally, let $\mathcal{D}_{\text{old}}$ be the dataset of the past task and $\mathcal{D}_{\text{new}}$ be the new task data. Then, the combined loss function becomes:

$$\mathcal{L}(\theta) = \lambda \mathcal{L}_{\text{old}}(\theta; \mathcal{D}_{\text{old}}) + (1 - \lambda)\mathcal{L}_{\text{new}}(\theta; \mathcal{D}_{\text{new}}) \tag{3.7}$$

where λ is a hyper-parameter that balances the importance between the old task and the new task.

3.3.2 Regularization-Based Methods

Regularization-based methods introduce penalty terms in the loss function to guide the learning of the new task in such a way that it prevents significant updates on those parameters that are important for the previous tasks. Two of the most common regularization techniques are:

- **Elastic Weight Consolidation (EWC)**: Uses the Fisher information matrix to identify and penalize changes in parameters that are important to old tasks (Kirkpatrick et al., 2016).
- **Learning without Forgetting (LwF)**: Uses the model's output on old tasks to prevent forgetting (Li & Hoiem, 2016).

In general, the regularized loss function for continual learning methods can be expressed as:

$$\mathcal{L}(\theta) = \mathcal{L}_{\text{new}}(\theta) + \lambda \mathcal{R}(\theta, \theta^*) \tag{3.8}$$

where $\mathcal{R}(\theta, \theta^*)$ is the regularization term that penalizes changes in the model parameters θ with respect to their previous values θ^*, and λ is a hyperparameter that balances the trade-off between learning new tasks and retaining previous knowledge.

3.3.3 Dynamic Architectures

Dynamic architecture approaches adjust the model's capacity by adding new parameters or modules for the new tasks, to allow the model to retain knowledge from previous tasks while incorporating new information.

An example of this strategy would be the *Progressive Neural Networks*, where the network grows by adding new columns for each task, with lateral connections to previously learned features (Rusu et al., 2016).

3.3.4 Complementary Learning Systems

Complementary Learning Systems (CLS) draw inspiration from the hippocampal and neocortical systems in the brain, combining rapid learning with slower, long-term knowledge consolidation (Kemker & Kanan, 2017; Mcclelland et al., 1995).

These methods often use a two-phase memory update mechanism: (1) a short-term memory system dedicated to capturing and processing immediate task information, and (2) a long-term memory system designed to consolidate and preserve essential knowledge over time. This dual structure ensures an effective balance between stability (retaining past knowledge) and plasticity (adapting to new information).

3.4 Continual Learning in NMT

Applying Continual Learning in NMT presents specific challenges due to the complexity of translating human languages, as well as specific approaches to deal with these problems.

3.4.1 Specific Challenges in NMT

Continual Learning in NMT comes with several challenges. First, these systems must learn intricate patterns in language, including syntax, semantics, and contextual nuances. Second, this learning must be done without forgetting previously acquired knowledge, as new linguistic information must be carefully added to avoid overwriting established language patterns or domain-specific terminology (Wu et al., 2024).

Another challenge lies in the open vocabulary problem. Given the fact that human languages are constantly evolving with new words and nuanced semantics, NMT models need to handle these changes without losing accuracy or performance. Simultaneously, these models must deal with the differences across human languages, balancing universal linguistic structures with language-specific variations, a task further complicated by the diversity of linguistic nuances across languages. Besides, limited computational resources add another layer of

complexity, as retraining models from scratch for every update is often infeasible on large models.

Finally, NMT encounters significant difficulties in addressing the data scarcity problem in low-resource languages, where the limited availability of high-quality data makes it challenging to update models without degrading their performance. Furthermore, it is essential to ensure that the biases inherent in the training data do not propagate or worsen over time, as continual updates can inadvertently amplify these issues.

3.4.2 Existing Approaches

Continual Learning has not been widely studied in NMT. As a result, most approaches tackle this problem indirectly or through the strategies discussed above, like fine-tuning models periodically, mixing new and old data to prevent forgetting (Robins, 1995), training a model on multiple language pairs to promote knowledge sharing and reduce forgetting (Arivazhagan et al., 2019), using separate modules for different languages or domains (Bapna & Firat, 2019).

Recent advances have explored modular approaches, where adapters or additional layers are dynamically added for new languages or domains (Bapna & Firat, 2019). These methods enable better retention of existing knowledge by isolating the changes to specific components while maintaining a shared backbone for universal features. Also, techniques such as EWC (Kirkpatrick et al., 2016) or other regularization methods (Carrión & Casacuberta, 2022b, 2024) have also been employed to penalize drastic parameter changes that could lead to forgetting.

Another promising approach involves leveraging large pre-trained multilingual models (Conneau et al., 2019; Liu et al., 2021), including Large Language Models (OpenAI et al., 2024; Zhu et al., 2024), as a starting point, fine-tuning them incrementally on new data. These models provide robust initial representations due to their extensive pretraining on diverse and multilingual datasets, reducing the risk of catastrophic forgetting (Winata et al., 2023). Similarly, generative replay methods are also used, where synthetic examples are generated to simulate past tasks (Shin et al., 2017).

However, despite these advances, the development of methods that can effectively balance the integration of new knowledge with the retention of existing one remains a major challenge in Artificial Intelligence (AI), especially in the field of NMT.

3.5 Evaluation of Continual Learning Methods

The evaluation of Continual Learning (CL) methods involves assessing both the model's ability to retain previously acquired knowledge and its capability to effectively incorporate new information (Lange et al., 2023; Soutif–Cormerais et al., 2023). In the context of NMT, this requires a combination of traditional evaluation metrics for translation quality and specialized metrics that capture the dynamics of continual learning.

In the following subsections, we will discuss these evaluation approaches, especially in the context of NMT.

3.5.1 Metrics and benchmarks specific to NMT

Evaluating Continual Learning in NMT requires metrics that assess both translation quality and knowledge retention (Cao et al., 2021; Garcia et al., 2021).

Traditional NMT metrics such as BLEU, ChrF, and TER, as discussed in Section 2.5, serve as foundational tools for translation quality assessment. However, continual learning presents specific challenges that require additional measures that take into account the retention and transfer dynamics.

To do so, we have complemented traditional NMT metrics with the following continual learning key measures (Rodríguez et al., 2018):

- **Backward Transfer (BWT)**: Quantifies the impact of learning new tasks on performance on previous tasks.
- **Forward Transfer (FWT)**: Measures how learning previous tasks helps in learning new tasks.
- **Forgetting Measure (FM)**: Calculates the difference in performance on previous tasks, before and after learning the new tasks.

3.6 Chapter Summary

This chapter has explored the concept of Continual Learning (CL) from both a theoretical and practical standpoint, emphasizing its importance in scenarios where systems must adapt to a continuous sequence of tasks while retaining previously acquired knowledge. We began by defining the CL problem within

the broader ML landscape, contrasting it with the traditional offline and static approaches, and highlighting its relation to online and incremental learning paradigms.

We then discussed three key challenges inherent to Continual Learning: (1) The Catastrophic Forgetting problem, where adapting models to new tasks disrupts previously learned representations; (2) Data distribution shifts, which occur when variations in data distributions degrade model performance over time; and (3) Scalability and computational constraints, which necessitate of memory- and computational-efficient solutions. These challenges are particularly critical in NMT, where continual learning enables NMT models to adapt dynamically to linguistic changes, such as incorporating new vocabulary terms, while maintaining accuracy across the different linguistic contexts.

Next, we surveyed popular continual learning strategies, including rehearsal methods, regularization-based techniques, dynamic architectures, and complementary learning systems. Each strategy aims to balance the plasticity needed to learn new tasks with the stability required to preserve existing knowledge. Consequently, we then examined how these strategies can be applied to NMT, identifying the specific challenges that arise from language diversity, domain shifts, open-vocabulary problems, and the need to integrate incremental information without retraining our models from scratch.

Finally, we outlined relevant evaluation methodologies, which combine standard NMT metrics such as BLEU, ChrF, and TER with specialized Continual Learning metrics to assess the retention of past tasks (backward transfer), the positive influence on future tasks (forward transfer), and the overall performance impact, often measured as FM. Together, these methods provide a framework to assess the effectiveness of Continual Learning approaches in NMT, enabling researchers and practitioners to benchmark how well these models can integrate new linguistic information while retaining their previously learned knowledge.

In summary, this chapter highlights the potential of continual learning approaches to advance the field of NMT, addressing key challenges and paving the way for more robust and adaptive solutions that align with the dynamic nature of human language.

Chapter 4

Experimental Framework

This chapter presents the experimental framework used in this work. It begins by detailing the datasets utilized, which span multiple data sources, languages, domains, and styles to ensure a comprehensive evaluation. Next, we describe the data processing techniques employed, including both pre-processing and post-processing steps that standardize and prepare the data for modeling. The chapter then discusses the modeling approaches, focusing on the selected architectures and the various vocabulary encoding strategies explored. We also outline the implementation details, including training procedures, hyperparameters, and the hardware setup used for the experimentation. Finally, we discuss the evaluation metrics used to assess the translation quality, justifying our choices to ensure a thorough and reliable study.

4.1 Data Sources

To evaluate our approaches in Continual Learning (CL) for Neural Machine Translation (NMT), we utilized a diverse set of datasets covering a wide range of languages, domains, and styles.

4.1.1 Description of training and evaluation datasets

To ensure a robust evaluation of our models across various linguistic and domain-specific scenarios, we utilized a diverse set of open-source datasets. Below is a brief description of each dataset:

- **Europarl Corpus**: Comprising parallel texts from the proceedings of the European Parliament (Koehn, 2005), this dataset is a rich source for formal and structured language pairs.
- **SciELO Corpus**: This corpus includes parallel sentences from scientific publications in the biomedical domain (Neves et al., 2016). A custom combination of its Health and Biological subsets, referred to as *SciELO (H+B)*, was created for this study.
- **Multi30K**: Originally consisting of image captions (Elliott et al., 2016), this dataset was later extended with a synthetically generated version, *Multi30K-Formality*, using the translation system developed by "DeepL Translator", 2023. This version introduces stylistic variations (neutral, formal, informal) for the *en-de/es* language pairs to explore stylistic variance in translations.
- **CommonCrawl**: A web-crawled corpus offering a diverse range of parallel sentences (Smith et al., 2013).
- **NewsCommentary**: This dataset contains parallel sentences from news and commentary articles (Tiedemann, 2012).
- **IWSLT'16**: Featuring parallel sentences from TED talks (Cettolo et al., 2012), this dataset is particularly suitable for evaluating models on spoken language and domain-specific challenges in speech translation.
- **JRC-Acquis**: A collection of legislative texts from the European Union, providing a reliable source of legal parallel data (Steinberger et al., 2006).
- **WikiMatrix**: Parallel corpora from Wikimedia (Schwenk et al., 2019).
- **Tatoeba**: A multilingual corpus consisting of crowdsourced translations (Contributors, 2013; Tiedemann, 2012).

In Table 4.1, we summarize these datasets, detailing the language pairs, domains, and training sizes used. Each dataset served a distinct purpose, such as exploring domain adaptation (e.g., SciELO), handling noisy web data (e.g., CommonCrawl), stylistic control (e.g., Multi30K-Formality), low-resource lan-

guages (e.g., Tatoeba), and more. Furthermore, the validation and test sets were often taken directly from the original datasets, although in some specific cases, small subsets (e.g., 1k to 5k sentences) were extracted from the training sets to create tailored validation sets (see corresponding papers).

Table 4.1: Summary of datasets used in the experiments. Values represent the number of sentences. Multiple training sizes indicate different subsets employed in the experiments. Validation and test sets corresponded to the original datasets, except in specific cases where 1k or 5k sentences were extracted from the training sets (see corresponding papers). Language pairs for Europarl included en-es, fr, de, cs, pt, it, sv, da, bg, zh, and ru.

Dataset	Languages	Domain	Train Size
Europarl	en-*	Parliamentary proceedings	∼50k/100k/1-2M
SciELO (Health)	en-es, pt	Biomedical	100k/120k
SciELO (Biological)	en-es, pt	Biomedical	100k/120k
SciELO (H+B)	en-es, pt	Biomedical	100k/240k
Multi30K	en-de	Image captions	29k
Multi30K-Formality	en-de, es	Synthetic Multi30K	29k
CommonCrawl	en-es	Web crawled data	100k/1.8M
NewsCommentary	en-de	News, Commentary	35k/357k
IWSLT'16	en-de	TED talks	196k
JRC-Acquis	en-es	Legal texts	100k
WikiMatrix	en-ar, ko, ja	Wikipedia	300-500k
Tatoeba	en-mr, or	Low-resource	10-15k

4.2 Data Processing

Most of the data processing steps were defined through the AutoNMT library (Carrión & Casacuberta, 2023a), a tool that we designed to streamline the research of sequence-to-sequence models by automating the pre/post-processing, training, and evaluation of NMT models.

4.2.1 Pre-processing Techniques

All datasets were preprocessed using standard techniques to ensure consistency and comparability across experiments. We utilized tools such as Sacremoses (Koehn et al., 2007; Technologies, 2020) for normalization and tokenization, as well as the Huggingface *Tokenizers* library (Moi & Patry, 2023). The pre-processing steps included:

- **Normalization**: Unicode normalization (NFKC), whitespace removal, and case folding (when applied)[1].
- **Tokenization**: In general, we made use of the SentencePiece tool (Kudo & Richardson, 2018). However, for word-level encodings, we processed all datasets using language-specific word-level tokenization using the Moses library (Koehn et al., 2007).

For vocabulary construction, we experimented with various encoding strategies: words, subwords, quasi-characters, characters, and bytes. Similarly, for subword-level encodings, we used a Byte-Pair Encoding (BPE) approach in most experiments. However, a few experiments were conducted using Unigram as a comparison, with no significant differences observed.

In addition, we usually remove duplicate source-target pairs and pairs with a significant difference in length between them.

4.2.2 Post-processing Techniques

After translation, the output sequences were detokenized and denormalized to restore the original text format for evaluation. That is, the post-processing steps were explicitly defined and designed to mirror the pre-processing techniques, applied in reverse order. Specifically, our post-processing includes:

- **Detokenization**: Reversing the tokenization applied during pre-processing. For subword-level tokenizations, we used the corresponding SentencePiece model, while for word-level tokenizations, we used Moses.
- **Denormalization**: Reverting any case folding or whitespace modifications made during pre-processing to align the output with the original input format.

Notably, no specific placeholder replacements were applied for elements such as URLs, dates, or numbers, following an end-to-end approach.

[1] Case folding was selectively applied, converting text to lowercase only in specific experiments to further reduce variability or training requirements.

4.3 Modeling Approach

This section provides an overview of the selected neural architectures, designed decisions, configurations, and vocabulary encoding strategies employed in this research.

4.3.1 Selected Model Architectures

In this section, we outline the key NMT architectures used in our experiments, focusing on their configurations, capabilities, and roles in our study.

While the Transformer architecture was the central focus of our work, we also included other prominent architectures such as Recurrent Neural Networks (Rumelhart et al., 1986), Bidirectional Recurrent Neural Networks (Sundermeyer et al., 2014), Gated Recurrent Units (Cho et al., 2014), Long Short-Term Memory networks (Hochreiter & Schmidhuber, 1997), and Convolutional Neural Networks (Conneau et al., 2016). These older architectures were incorporated into a few very specific experiments to provide comparative baselines and assess the impact of architectural differences on the proposed methods. In Table 4.2, we summarize the main characteristics of the model architectures used, including parameter ranges and hyperparameter configurations.

Table 4.2: NMT architectures used in our research: The number of parameters varied depending on the vocabulary size used in each configuration.

Model	Parameters	Hyperparameters
LSTM	7.6M-11.5M	3 layers / 256 emb / 256 hid / bidir+attn
GRU	7.0M-80.2M	3 layers / 256 emb / 512 hid / bidir+attn
Fully CNN	33.0-35.9M	10 layers / 256 emb
Transformer Small	4.1M-25.0M	3 layers / 8 heads / 256 dim / 512 FFN
Transformer Standard	44.9M-92.7M	6 layers / 8 heads / 512 dim / 2048 FFN

The Transformer architecture was chosen as the principal model for its superior performance in both accuracy and training efficiency (Kocmi et al., 2023; Vaswani et al., 2017). Additionally, smaller variants of the Transformer (e.g., Transformer Small) were employed to accelerate the research iteration, as we noted that during most of our experimentation, they achieved a performance comparable to the larger variants.

To ensure a comprehensive and fair analysis, the other architectures were used in scenarios where their distinct characteristics (e.g., recurrent structures in LSTMs and GRUs or the depth of CNNs) could provide some valuable

insights into the challenges and benefits of the continual learning methods that we proposed. These experiments were conducted under carefully controlled settings to ensure the comparisons remained as fair and consistent as possible.

4.3.2 Vocabulary Encoding Strategies

To address the open-vocabulary problem and the catastrophic forgetting problem, we explored several vocabulary encodings as discussed above. However, these encodings were applied before the training process, meaning that all NMT models, regardless of the encoding used, processed their inputs/outputs as a sequence of tokens separated by whitespace.

4.4 Implementation Details

This subsection provides a comprehensive overview of the implementation details, highlighting key aspects of the experimental setup, training configurations, and hardware environments used throughout this work.

4.4.1 Hyperparameters

Training hyperparameters were carefully selected to ensure fair comparisons across our experimentation. However, due to the extensive range of experiments conducted, listing the exact hyperparameters for each one of them is impractical (for these specific details, please refer to the published papers). Nonetheless, we used common hyperparameters that achieved highly consistent results regardless of the specific values. These hyperparameters include:

- **Loss Function**: Cross-Entropy loss was employed, often with label smoothing values between 0.1 and 0.2, to mitigate overconfidence in model predictions.
- **Optimizer**: Most experiments utilized *Adam* (Kingma & Ba, 2015) or *AdamW* (Loshchilov & Hutter, 2017), with initial learning rates ranging from $1e^{-3}$ to $1e^{-5}$, and a warm-up phase of 4,000 steps was included for stable convergence. Besides, for fine-tuning tasks, a *SGD* (Robbins & Monro, 1951) with momentum (0.9) was used.
- **Batch Size**: A batch size of 128 sentences or 4,096 tokens was utilized and adjusted based on the model size, GPU memory constraints, and library version (features supported).

- **Learning Rate Scheduling**: Default scheduler settings from the chosen optimizer were applied, with minor adjustments for specific experiments to improve stability.
- **Gradient Clipping**: Gradient norms were clipped at 0.1 or 1.0 to prevent instability caused by exploding gradients.
- **Maximum Epochs**: Experiments were run for 50 to 200 epochs, with early stopping (patience of 10 or 15 epochs) based on validation loss or BLEU score.
- **Beam Search**: Beam widths of 1 to 5 were explored during the decoding phase, with higher values primarily used for translation quality evaluations.

4.4.2 Training procedures

Training was performed across different hardware and environments to evaluate scalability and reproducibility:

- **Hardware configurations:** Experiments were performed on consumer, high-performance NVIDIA GPUs, including:
  - Dual NVIDIA TITAN Xp GPUs with 12GB VRAM each.
  - NVIDIA RTX 2080 GPUs with 8GB VRAM.
  - NVIDIA RTX 4090 GPUs with 24GB VRAM.
- **Software Environments:** All training processes ran on Ubuntu-based systems with CUDA 11.0 or higher and the corresponding cuDNN libraries.
- **Frameworks:**
  - **Fairseq** (Ott et al., 2019): This framework was used during the initial stages of our experimentation and later utilized as a reference toolkit to validate the results and ensure the reliability of our custom research toolkit.
  - **AutoNMT** (Carrión & Casacuberta, 2023a): A Python library that we designed to streamline the research of sequence-to-sequence models by automating pre-/post-processing, training, and evaluation of NMT models.

To minimize variance in training outcomes:

- **Shuffling and Balancing:** Training data was shuffled at the start of each epoch. In multilingual tasks, datasets were balanced to provide equal exposure across all languages or domains, preventing overexposure bias during training.
- **Tagging:** Input sequences in multilingual and multi-domain setups were prefixed with tags to distinguish between languages, domains, or stylistic variations.
- **Continual Learning Protocols:** For continual learning experiments, data sequences followed a predefined order. This ensured consistent evaluation of model retention and catastrophic forgetting across different experimental setups.

To optimize computational resources:

- Mixed-precision training (FP16) was employed to accelerate training and reduce memory requirements, particularly for larger Transformer models.
- Checkpointing was performed at the end of each epoch, with redundant checkpoints periodically deleted to save disk space.
- Multi-GPU setups were utilized for parallel model evaluation, significantly reducing the total training time for iterative tuning processes.

4.5 Evaluation Metrics

Although these metrics were previously introduced in Chapter 2.5, this section revisits and contextualizes their use within the experimental framework used to evaluate our translation models. This redundancy highlights the specific rationale behind their selection, bridging the theoretical concepts with their practical application in our experimentation.

4.5.1 Justification for chosen metrics

We used automatic evaluation metrics to quantitatively assess the translation quality of our models:

- **BLEU** (Papineni et al., 2002): This metric is the most widely used metric in machine translation and the defacto metric in our work.
- **chrF** (Popović, 2015): This metric was sensitive to morphological variations and suitable for languages with rich morphology.
- **Translation Error Rate** (Snover et al., 2006): This metric provides insight into the efforts required to correct translations.
- **BERTScore** (T. Zhang et al., 2019): This metric captures meaning beyond surface forms due to the use of contextual embeddings.

While we primarily reported BLEU scores for clarity, we also used other metrics to validate the consistency of our results. For this case, SacreBLEU was chosen to ensure that our evaluation scores were comparable and reproducible with the official WMT scores, as it standardizes the tokenization and calculation parameters (Post, 2018).

4.6 Chapter Summary

In this chapter, we have described the experimental framework that supports our research on Continual Learning in Neural Machine Translation. First, we describe the datasets utilized, covering multiple languages, domains, and styles to ensure a robust evaluation of the proposed methods. Then, we detailed the data processing pipeline, including both pre-processing and post-processing steps, which together help to maintain the consistency of our experimentation.

Next, we discussed our modeling approach, focusing on the selection of NMT architectures and vocabulary encodings. Although the Transformer was our primary architecture due to its State-of-the-Art performance, we included other neural architectures, such as RNNs, LSTMs, GRUs, and CNNs, to provide a strong comparative baseline.

In addition to this, we also provided information about the implementation details, including hyperparameters, hardware used, and training procedures, to ensure a fair comparison of methods during our research.

Finally, we revisited our choice of evaluation metrics, providing a justification for each one based on their ability to capture specific aspects of translation quality.

Chapter 5

Addressing the Open-Vocabulary Problem in NMT

This chapter discusses the challenges posed by an open vocabulary in Neural Machine Translation (NMT) systems. As language continuously evolves, new words and expressions are constantly introduced, presenting a unique problem in maintaining high-quality translation output without requiring extensive retraining. Various encoding strategies, their trade-offs, and approaches to manage open-vocabulary challenges are explored.

5.1 The Open-Vocabulary Problem

The open-vocabulary problem in NMT arises from the ever-evolving nature of language, where new words emerge continuously. Moreover, even with large training datasets, these problems will still persist, as no matter how large and extensive a dataset is, it will always be finite. Consequently, some words may inevitably never appear during training, leaving the model unable to generate reliable translations for these unseen terms at inference time. This creates a challenge for models that operate on fixed vocabularies. The inability to handle Out-of-Vocabulary (OOV) words effectively can lead to degraded translations. This section discusses the core issues related to this problem and examines different vocabulary encoding strategies to determine whether any vocabulary type inherently offers superior performance.

5.1.1 Understanding the Open-Vocabulary Problem

Language is constantly evolving, with new words and phrases at a rapid pace. However, NMT systems typically rely on fixed vocabularies defined at training time. This limitation makes it difficult for models to handle OOV (out-of-vocabulary) words effectively, especially when encountering new terms or domain-specific expressions (T. Luong et al., 2014; Sennrich et al., 2016). Consequently, translation quality can decline, particularly in specialized fields or low-resource languages where the lexicon can expand quickly (Koehn & Knowles, 2017; Sato et al., 2020).

The inability to seamlessly incorporate new terminology not only hinders the model's ability to produce accurate translations but also hampers its generalization abilities. As a result, when confronted with OOV words, these models may resort to undesirable coping strategies such as producing placeholder tokens, dropping the unknown word, or mistranslating it, thus, leading to a significant degradation in the overall performance (Fernando & Ranathunga, 2022). In a continual learning context, where NMT models must adapt to new data over time without forgetting previously learned information, the open-vocabulary challenge becomes even more prevalent (Cao et al., 2021).

Because of that, ensuring that NMT models can continuously integrate new words without an extensive retraining process is essential for maintaining a robust generalization across an ever-evolving landscape of languages and domains.

5.1.2 Vocabulary Encoding Strategies

In this section, we examine various vocabulary encoding strategies, considering the trade-offs they present in terms of vocabulary size, information density, and model complexity. By doing so, we aim to determine if any particular encoding approach is inherently more suitable for NMT, especially within the context of continual learning.

Comparison of vocabulary types

As discussed in Section 2.3.1, vocabulary encodings in NMT can generally be categorized into four levels: byte-level, character-level, subword-level, and word-level. Each type offers distinct advantages and trade-offs concerning vocabulary size, sequence length, and handling of OOV words. However, subword-level encodings can be seen as a hybrid approach, bridging the gap between low-level

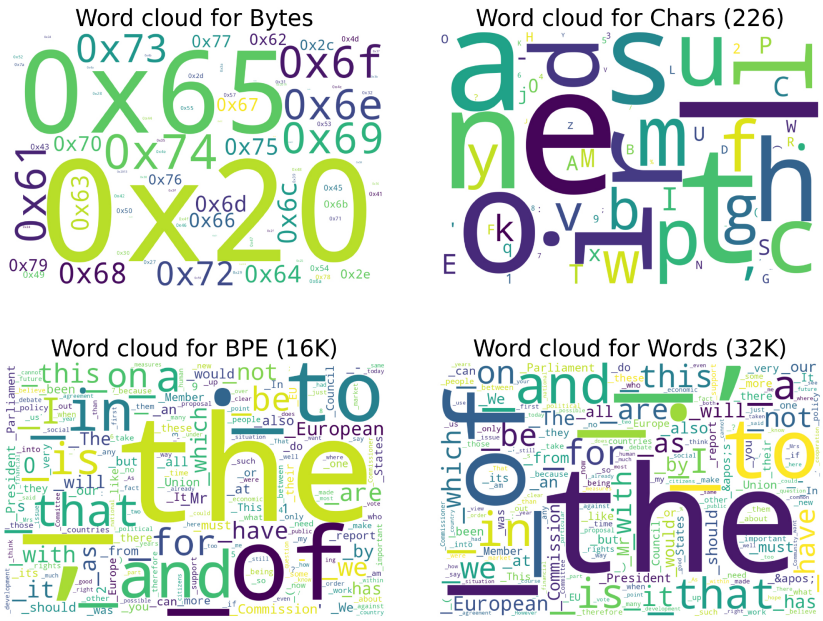


Figure 5.1: Word clouds of different vocabulary types: (Top-left) Byte-level encoding; (Top-right) Character-level encoding; (Bottom-left) Subword encoding using Byte-Pair Encoding (BPE); and (Bottom-right) Word-level encoding

units (e.g., bytes, characters) and higher-level units (i.e., words). That is, they interpolate between the flexibility and OOV-handling capabilities of lower-level encodings with the shorter sequence lengths and higher information densities of typical word-level representations.

- **Byte- and character-level encodings** offer finer granularity and allow NMT systems to handle OOV words more effectively. However, they tend to generate longer sequences, which can increase the computational burden and make it harder for models to learn long-term dependencies.
- **Subword-level encodings** strike a balance between flexibility and efficiency by breaking words into smaller, commonly occurring units. This reduces the OOV problem while keeping sequence lengths manageable.
- **Word-level encodings**, while straightforward, result in large vocabularies and are prone to OOV issues, especially in languages with rich morphology or when handling low-resource datasets.

The choice of encoding can significantly influence the model's ability to generalize, learn new words continually, and adapt to changes in the training data over time. Therefore, we ask ourselves the question: *Is there any vocabulary inherently better?*

To answer this, we evaluate these encoding strategies from four key perspectives:

- **Grammatical Equivalence:** From a theoretical standpoint, any vocabulary encoding, whether based on characters, subwords, or words, can represent the same grammatical structures (P. F. Brown et al., 1993; Chomsky, 1956). This is because valid linguistic sequences can always be reconstructed from lower-level encodings or through an infinite high-level vocabulary (Sennrich et al., 2016; Sutskever et al., 2014). However, the efficiency of these encodings hinges on striking a balance between the token granularity and the model's ability to capture grammatical dependencies effectively (Belinkov et al., 2019; Gowda & May, 2020).

- **Information Density:** Information density refers to how much linguistic information is captured by each token in an encoding (Shannon, 1948). Finer-grained encodings, such as byte- or character-level vocabularies, typically result in more tokens per sentence, reducing the information each token carries (Neubig et al., 2013). This, in turn, places a greater burden on the model to learn relationships across many more tokens. In contrast, subword and word-level encodings concentrate more information into fewer tokens, potentially allowing the model to learn more efficiently with shorter sequences (Sennrich et al., 2016).

- **Long-Term Dependencies:** Longer sequences, commonly associated with byte- and character-level encodings, pose a significant challenge for models to effectively capture long-term dependencies. Moreover, these encodings often require substantially more computational resources (Hochreiter & Schmidhuber, 1997; Vilar et al., 2007). This issue is particularly pronounced in Transformers, where computational costs scale quadratically with the sequence length, making fine-grained encodings both expensive and slow to train (Vaswani et al., 2017). In contrast, subword- and word-level encodings result in shorter sequences, enabling models to focus on the most relevant information within shorter and more manageable context (Sennrich et al., 2016; Vaswani et al., 2017).

- **Long-Tail Distributions:** In natural languages, word usage often follows a long-tail distribution, where a few words are very common, but many words or subword units are rare. Byte- and character-level encodings han-

dle rare words well, since they can decompose any word into manageable tokens. Subword approaches offer a compromise, covering common words with fewer tokens while still being flexible enough to handle rare words via smaller subword units. Word-level vocabularies, however, struggle with these rare words, exacerbating OOV issues in low-resource or continually evolving datasets (Raunak et al., 2020).

Grammatical equivalence and compositionality

Our first claim revolves around the concept of Grammatical Equivalence (P. F. Brown et al., 1993; Chomsky, 1956), wherein we state that:

> *In theory, no vocabulary encoding is inherently superior to another if both encodings are grammatically equivalent.*

While this holds in theory, further clarification is needed. Namely, drawing an analogy to the well-known *no free lunch* theorem (Wolpert & Macready, 1997) in optimization theory, we argue that no encoding can be universally optimal. Because even with a sufficiently large and grammatically equivalent vocabulary, practical differences may arise as some encodings provide better granularities at the expense of more computational or memory requirements. Additionally, practical challenges in managing the OOV words with these finite vocabularies can impact both the grammatical equivalence and the overall effectiveness. Thus, we argue that no single encoding strategy is universally optimal across all tasks, datasets, and computational constraints.

To further explore how these theoretical considerations play out in practice, we conducted an empirical analysis across multiple languages and datasets to quantify the performance of different encoding strategies in capturing linguistic structures while considering the trade-offs they introduce in handling unknown tokens and preserving compositionality. The tables below summarize key statistics derived from encoding the English and Spanish training sets of the Europarl dataset (see Table 5.1). Similar results were observed across other languages and datasets.

From these results, we observe several key trends:

- **Byte- and character-level encodings** result in a higher number of tokens per word and per sentence due to their fine granularity. This granularity ensures that all words, including unseen or rare ones, can

Table 5.1: Vocabulary statistics for English and Spanish in the Europarl dataset. The *Unknowns per 1M Tokens* column quantifies Out-of-Vocabulary (OOV) severity.

Language	Vocabulary Type	Tokens/Word	Tokens/Sentence	Unknowns per 1M Tokens
English	**Bytes**	5.95	148.92	0.0
	Characters	5.99	149.92	0.16
	BPE (8k/16k/32k)	1.25 / 1.18 / 1.15	31.19 / 29.57 / 28.90	0.86 / 0.84 / 0.79
	Words (8k/16k/32k)	1.00	27.72	28,525 / 12,073 / 4,602
Spanish	**Bytes**	6.14	161.26	0.0
	Characters	6.18	162.26	0.14
	BPE (8k/16k/32k)	1.29 / 1.20 / 1.16	33.93 / 31.59 / 30.43	0.77 / 0.74 / 0.69
	Words (8k/16k/32k)	1.00	28.98	50,637 / 25,368 / 11,062

be represented without introducing unknown tokens. As a result, these encodings fully preserve compositionality and grammatical completeness, allowing for the reconstruction of any word.

- **Subword encodings** strike a balance between granularity and sequence length, reducing the number of tokens per word while maintaining a low rate of unknown tokens. This makes them particularly efficient in handling unseen, rare, or morphologically complex words by recomposing them from familiar subword units and without excessively increasing sequence length.

- **Word-level encodings**, while intuitive and straightforward, they suffer the highest rate of unknown tokens due to their lack of compositionality capabilities that would allow them to break down words into smaller compositional units. This limitation significantly degrades the model performance, especially in low-resource or open-domain settings where new or rare words frequently occur. As a result, these vocabularies are poorly suited for tasks that demand high adaptability, such as continual learning, where the ability to handle evolving vocabularies is crucial.

In summary, while no vocabulary type is inherently superior from a grammatical standpoint, practical factors play a critical role in determining their effectiveness such as the handling of unknown words, symbols, and computational efficiency. Subword encodings, especially when combined with byte-level fallback mechanisms, offer an excellent balance by preserving compositionality, efficiently managing rare and unknown words, and maintaining manageable sequence lengths. This makes them a favored choice for most NMT applications. However, subword units can be highly language-dependent unless trained on large, multilingual datasets, and scaling this approach across hundreds of languages

may introduce a significant computational overhead. On the other hand, byte- and character-level encodings excel in scenarios requiring language-agnostic solutions, full compositional fidelity, and robustness against OOV words. Specifically, these properties are particularly interesting for low-resource settings, where the subword units may not provide the high-quality representations needed for effective lifelong translation systems.

Information density

Our second claim revolves around the concept of Information Density (Shannon, 1948), which refers to the amount of information each token carries within a given sentence, or in other words:

> *The more tokens we use to encode a given sentence, the less information each token conveys.*

While this may seem intuitive, explicitly studying information density is crucial to quantify its impact, as encoding strategies often yield to distinct sequence lengths, affecting computational costs, model complexity, and the ability to capture long-term dependencies. Accordingly, encoding strategies that use more tokens to represent a given sentence often convey less information per token. This is because each token in a low-level encoding, such as bytes or characters, captures a smaller fragment of the linguistic information compared to tokens in subword- or word-level encodings (Costa-jussà et al., 2017; Vilar et al., 2007).

In the context of NMT, this concept plays a critical role in balancing model efficiency and its complexity. This is because high-information density vocabularies (such as word-level tokens) enable shorter sequences, simplifying the task of capturing grammatical and semantic structures. However, encoding schemes with low information density (like byte- or character-level tokens) tend to create longer sequences, which increases the computational requirements and make the learning of long-term dependencies more difficult. On the other hand, these fine-grained tokens can more flexibly handle OOV words, especially in low-resource scenarios.

To better understand the impact of different vocabulary encoding strategies on the information density topic, we studied the relationship between the *average number of tokens per sentence* as a function of the *vocabulary size*, using a subword encoding (BPE), and a char encoding as a reference. These encodings applied in multiple languages and domains, as illustrated in Figure 5.2 and Figure 5.3 below.

For our first experiment, we studied the impact of different languages on the information density topic.

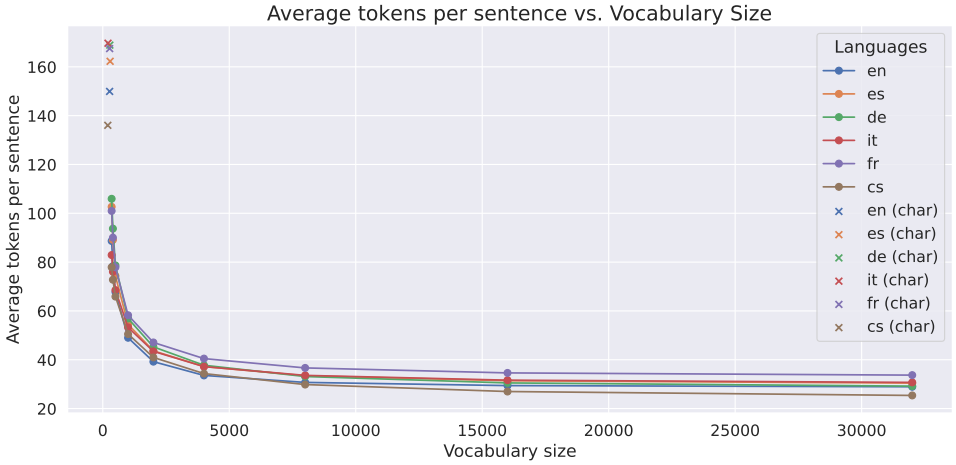


Figure 5.2: Average tokens per sentence as a function of vocabulary size for different languages: English (blue), Spanish (orange), German (green), Italian (red), French (purple), and Czech (brown). All from the Europarl dataset. The results show an exponential reduction in the number of tokens when transitioning from character-based sets to subword vocabularies, particularly up to around 5,000 tokens.

In Figure 5.2, we can see the relationship between the *Average tokens per sentence* as a function of the *Vocabulary size*[1] for six languages (English, Spanish, German, Italian, French, and Czech), from the Europarl dataset. The solid lines represent subword-level encodings (specifically, BPE), while the "x" markers indicate character-level encodings. As it can be seen in the figure, subword-level vocabularies exponentially decrease the number of tokens needed to encode a given sentence as their vocabulary size increases. In contrast, character-level and small subword-level vocabulary (more on this later) yield to longer sequences, therefore reducing the amount of information that each token conveys.

Interestingly, as the subword vocabulary size decreases, the sequence length starts to resemble that of a character-based encoding, suggesting that subword encodings can be viewed as a sort of continuous spectrum rather than a discrete set of encoding categories. Similarly, as the size of subword-level vocabularies

[1]The vocabulary size refers to the number of distinct tokens (words, subwords, or characters) that the NMT model can recognize and process during training and inference.

increases, particularly around 5,000 tokens, we can see that the average number of tokens per sentence tends to stabilize, resulting in a more word-like encoding.

This phenomenon was consistent across different languages, demonstrating how subword-level vocabularies efficiently compress token sequences, resulting in fewer tokens required per sentence, thus maintaining a higher information density than low-level encodings. Additionally, subword-level encodings offer a convenient flexibility, allowing it to shift towards a more character-based behavior or a more word-based representation, depending on the vocabulary size.

Next, we repeated the previous experiment but, this time, focusing on the impact of different domains (datasets) on the information density: *Europarl*, *SciELO (Health)*, and *JRC Acquis (legal)*.

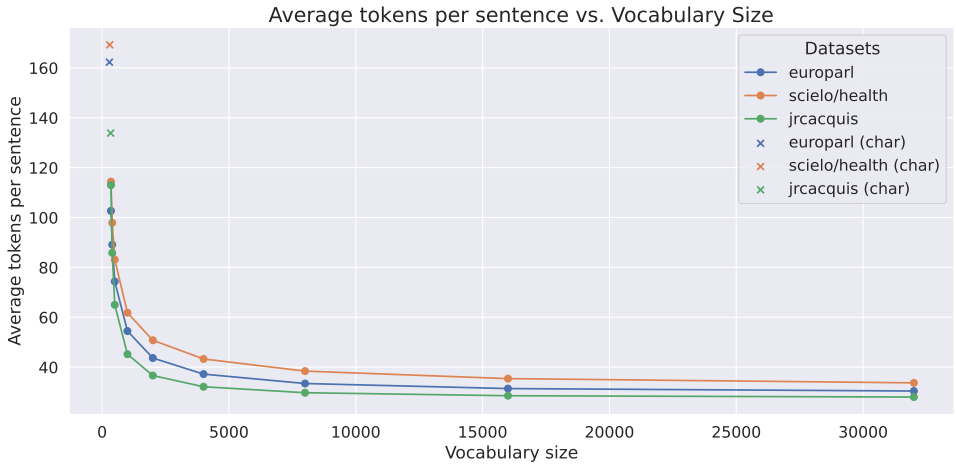


Figure 5.3: Average tokens per sentence as a function of vocabulary size for different domains: Europarliamentary (blue; Europarl), Biomedical (orange; Scielo/Health), and Legal (green; JRC-Acquis). The results show an exponential reduction in the number of tokens when transitioning from character-based sets to subword vocabularies, particularly up to around 5,000 tokens.

As shown in Figure 5.3, the patterns observed for the different domains are consistent with those seen across the different languages (see Figure 5.2). While smaller vocabularies resulted in significantly longer sequences, reflecting a lower information density per token. However, as the vocabulary size was increased, the sequences started to become more efficient, with fewer tokens needed to represent each sentence, thereby improving the overall information density.

The analysis of information density in both cases presents minimal variability and characterizes an important trade-off in vocabulary encoding strategies, which we expect to easily extrapolate to other scenarios. However, the main finding here is that subword-level vocabularies can be seen as an adaptive vocabulary encoding method that bridges the gap between character-based and word-based representations, offering a continuous spectrum of vocabulary densities. Therefore, by adjusting the vocabulary size, subword units can result in low-density encodings like characters or high-density encodings like words. This adaptability is particularly interesting, as it enables a more precise control over the token granularity and allows us to study the effects of the vocabularies more carefully.

The analysis of information density across both languages and domains shows minimal variability and characterizes an important trade-off in vocabulary encoding strategies. The main insight from this section, is that subword-level vocabularies offer a convenient and flexible encoding that can operate in a continuum spectrum, ranging from low-density encodings (like characters or quasi-characters) to high-density encodings (like words). This flexibility is particularly interesting, as it enables a more precise control over the information density trade-off, making it easier to examine how different encoding strategies affect a variety of Natural Language Processing (NLP) tasks or problems, such as the Continual Learning problem in NMT.

Long-term dependencies and long-tail distributions

One of the critical challenges when choosing vocabulary encodings in NMT lies in managing long-term dependencies and long-tail distributions (Hochreiter & Schmidhuber, 1997; Raunak et al., 2020). Our analysis reveals that models trained on low-level vocabularies, such as byte- or character-based encodings, tend to struggle with learning long-term dependencies due to the extended sequence lengths they generate. In contrast, higher-level vocabularies, like subword or word-level encodings, condense these sequences, making it easier for models to capture relationships between distant tokens. However, these higher-level vocabularies face significant challenges related to long-tail distributions, which affect their ability to learn from rare tokens.

We summarize this observation in the following claim:

> *The handling of long sequences produced by low-information density vocabularies requires of larger models with greater generalization capabilities.*

To illustrate these points, we conducted an experiment comparing two identical Transformer models trained on the Multi30K dataset (German-English). The only difference was the vocabulary encoding: one model used a character-based encoding, while the other used a word-based encoding. Our goal was to visualize how these different encoding strategies impact attention mechanisms.

In Figure 5.4, we observe the attention maps captured during the translation of a German sentence to English. Both models exhibit a diagonal pattern in the attention map, which is expected, as German and English have similar sentence structures, and the attention mechanism in Transformer architectures typically aligns corresponding words or characters between the two languages. However, the character-based model (left) shows a more scattered diagonal. This can be attributed to two main factors. First, the attention map shown corresponds to a single attention head out of the eight available, each capturing subtle variations in attention patterns. Second, it reflects the additional challenges posed by longer sequences in character-level encodings. It is worth pointing out that even though the granularity of the axes might differ (i.e., characters vs. words), this scattered pattern is consistent across all experiments, sentences, and attention-heads visualized. Hence, this behavior aligns with the expectation that character-level models must attend to more tokens simultaneously, resulting in a lower relative pressure on individual tokens under the same training conditions. Consequently, the increased sequence length in the character-based model makes it more challenging to efficiently learn long-term dependencies, as the model must process significantly more tokens compared to the word-based model.

Moreover, it is important to note that the computational complexity of the attention mechanism of the Transformer architecture is quadratic with respect to the sequence length, which means that low-information density models (i.e., chars) will consume significantly more memory and computational resources than their high-information density counterparts (i.e., words). As the transformer architecture is the most widely used architecture nowadays, this higher consumption creates a trade-off between the flexibility provided by lower-level vocabularies and the memory footprint required to process these longer sequences. In addition to this, we repeated this kind of visualization with other languages and domains, observing similar results.

Therefore, while low-level encodings offer greater flexibility in handling OOV words and linguistic variations, the challenge of learning long-term dependencies over extended sequences introduces a significant bottleneck. In contrast, word-based models can capture these dependencies more easily due to their shorter sequences, but they must address issues arising from long-tail distributions,

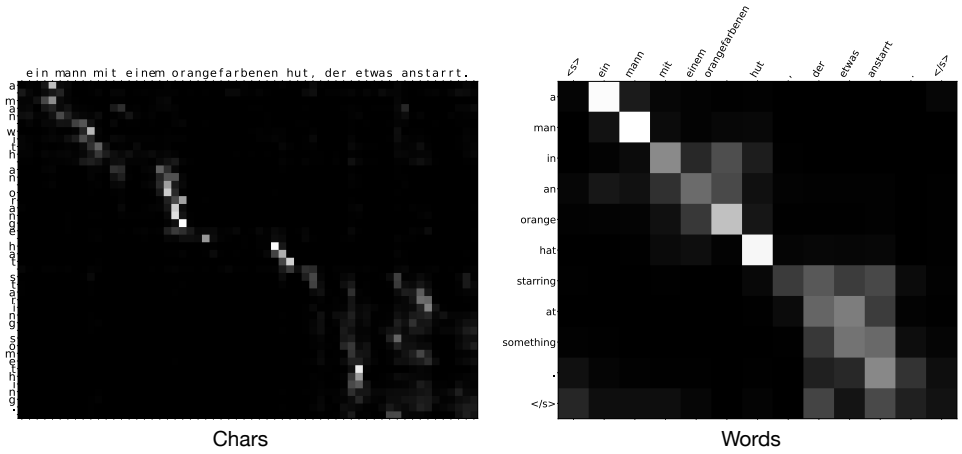


Figure 5.4: Attention heat maps for character- (left) and word-based (right) Transformer models: These figures illustrate the results from training a specific German-to-English NMT model, serving as an example of the trends observed across various languages and domains. Notably, despite the effectiveness of the Transformer's attention mechanism, capturing long-term dependencies becomes increasingly challenging as the sequence length grows (see scattering on the left figure). Thus, this effect is particularly pronounced in models using low-level encodings.

where less frequent tokens (such as rare words or subwords) may be harder for the model to learn effectively.

In natural languages, word frequency typically follows a long-tail distribution: a few words are highly frequent, while most words are rare. High-level vocabularies, such as word- or subword-based encodings, often suffer from this issue. Tokens in the tail of the distribution are used so infrequently that the model struggles to learn robust representations for them, especially in low-resource scenarios. This leads to a low utilization rate for many tokens in the tail, reducing the model's ability to generalize effectively.

Low-level vocabularies, such as byte- or character-based encodings, naturally mitigate this problem. By decomposing words into smaller units, these encodings distribute token frequencies more evenly, ensuring that rare words can still be represented through commonly occurring tokens. This leads to a more balanced frequency distribution, allowing the model to learn high-quality representations for infrequent tokens.

To explore this further, we analyzed the token frequency distributions of four types of vocabularies: byte-level, character-level, subword-level, and word-level encodings. In Figure 5.5, we can see the frequency distribution of the top 50 most frequent tokens, while Figure 5.6 shows the complete distribution for all tokens (zoomed in for BPE and words). These encodings were done for the English language in the Europarl dataset. Also, we repeated it for other languages (i.e., Spanish, German, Czech,...) and datasets (i.e., CommonCrawl, NewsCommentary, SciELO,...) with similar results.

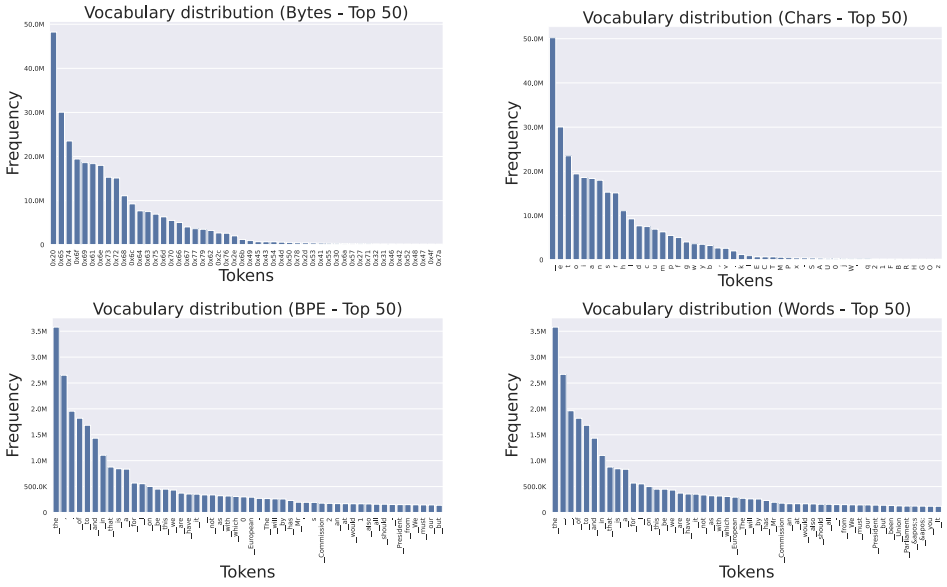


Figure 5.5: Frequency distribution of the top 50 tokens: (top-left) Byte-level, (top-right) Character-level, (bottom-left) Subword-level (BPE), and (bottom-right) Word-level vocabularies. The top 50 tokens display an apparently consistent frequency distribution across all encoding strategies. However, as we extend the analysis to include the full distribution, their shape becomes significantly different (see Figure 5.6).

As shown in Figure 5.5, the token distribution for the top 50 tokens appeared to be remarkably similar across all vocabulary types. However, after examining the full token frequency distribution, shown in Figure 5.6, we could observe a distribution pattern consistent with Zipf's law. This law states that word frequency in natural languages follows a power-law distribution, where common words dominate usage while the majority are rare. Consequently, all encoding distributions reflected this disparity, hinting that such token distributions are likely to generalize across human languages.

However, given the fact that the size of the vocabulary set varies with the encoding type, this leads to some significant differences in the shape of the long-tail distribution. That is, encodings such as subword- and word-level will exhibit longer tails compared to byte- and character-level encodings, resulting in varying degrees of token sparsity. This skewed token usage poses a significant challenge for NMT models, as they struggle to learn effective representations for infrequent or rare tokens. The issue is especially pronounced in low-resource settings, where the limited data exacerbates the difficulty of capturing meaningful patterns for these tokens.

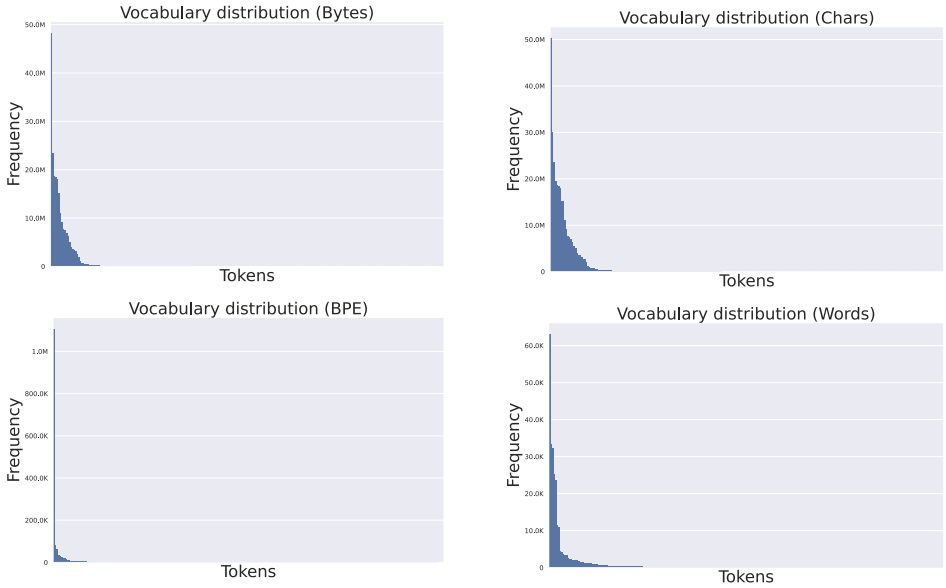


Figure 5.6: Full token frequency distribution: (top-left) Byte-level, (top-right) Character-level, (bottom-left) Subword-level (BPE), and (bottom-right) Word-level vocabularies. The shape of the full distribution varies significantly across encoding strategies, leading to notable high variations in the token usage rates depending on the chosen encoding. Notes: 1) The y-axis is zoomed in for BPE and Word-level vocabularies to highlight differences in frequency distribution. 2) Token labels are omitted for clarity due to the impracticality of displaying them.

In contrast, byte- and character-level encodings exhibit a higher frequency profile since, by decomposing rare words into smaller, more commonly occurring components, they ensure that the model is exposed to each token (or sub-token) more frequently (in absolute terms). This increased exposure leads to more stable representations, even for words that appear only rarely. Thus,

and reflecting our earlier observations that subword-level encodings can form a continuous spectrum between character- and word-level approaches (see Section 5.1.2), in Section 5.2 we will explore how quasi-character-level encoding might bridge the gap between these two paradigms.

In summary, while higher-level vocabularies (e.g., word-level) may offer efficiency in terms of sequence length and information density, they are particularly vulnerable to the long-tail distribution of token frequencies and struggle to handle rare words effectively. On the other hand, lower-level vocabularies (e.g., character-level) provide a more balanced representation of token frequencies, thus better managing rare words, even though this often comes at the cost of longer sequences. These considerations are especially important in low-resource or continual learning scenarios, where the handling of infrequent tokens is crucial.

5.1.3 Trade-Offs in Vocabulary Design

Designing an effective vocabulary requires balancing multiple factors, including data volume, vocabulary complexity, and model generalization.

In the previous section 5.1.2, we have discussed the theoretical implications of each vocabulary encoding. Now, we discuss the practical implication of these vocabularies in terms of performance.

Data volume vs. Vocabulary complexity vs. Model generalization

Larger vocabularies require more data to train effectively, as the model needs to learn representations for a wider variety of tokens. However, overly complex vocabularies can hinder generalization, particularly in low-resource settings. This trade-off must be carefully managed to ensure that the model can generalize while maintaining a manageable vocabulary size.

Trade-offs in vocabulary design

In this section, we analyze the trade-offs between different vocabulary encoding strategies by comparing their performance across varying amounts of training data. This analysis builds upon prior works such as Mielke et al., 2021, X. Zhang and LeCun, 2017, and Gowda and May, 2020, but with a specific focus on the challenges associated with the NMT problem. Specifically, we trained a set of Transformer models on the Europarl English-Spanish dataset, using

four distinct vocabulary encodings (bytes, chars, subwords, and words), three subsets of it, with 50k, 100k, and 200k sentences each one, in order to represent low-resource, mid-resource, and high-resource scenarios.

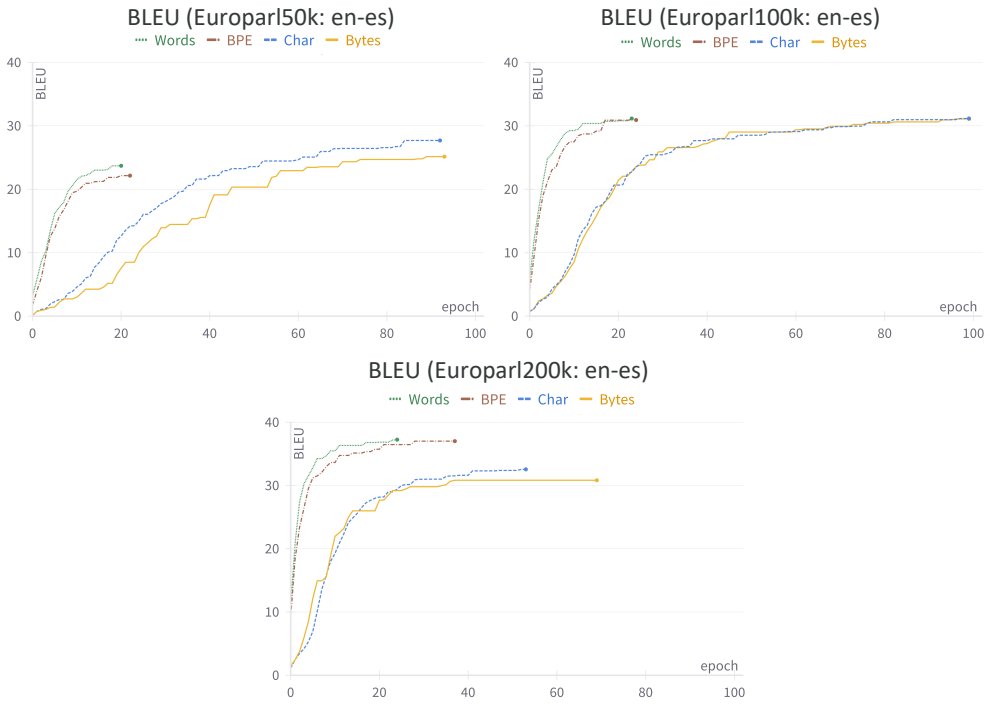


Figure 5.7: Comparison of vocabulary encodings by training volumes (50k, 100k, 200k sentences): Low-level encodings (byte and character) outperform high-level encodings (BPE and word) in low-resource scenarios, while high-level encodings achieve superior performance in high-resource scenarios.

As shown in Figure 5.7, this experiment shows that the effectiveness of each encoding strategy is heavily dependent on the amount of training data.

In the low-resource scenario (i.e., 50k sentences), the low-level encodings (byte and chars) achieved a higher performance than high-level encodings (subwords and words) due to their higher token utilization and compositional capabilities.

Then, in the mid-resource scenario (100k sentences), the performance across all encoding strategies converged, indicating that both low- and high-level encodings were able to capture the necessary linguistic relationships to achieve similar translation quality.

Finally, in the high-resource scenario (i.e., 200k sentences), the high-level encodings outperformed the low-level ones. This is in line with the previous theoretical discussion, as higher-level vocabularies are expected to capture semantic and syntactic patterns more efficiently if there is enough data. Furthermore, these high-level encodings benefit from the shorter sequence lengths, as their complexity is reduced compared to the longer sequences generated by low-level encodings. Therefore, this allows the model to learn long-term dependencies more effectively.

However, across all data volumes, low-level encodings exhibited slower convergence rates compared to high-level encodings (around 2 or 3 times slower), probably due to a combination of the smaller batches (same tokens, less sentences), their lower information density and longer sequences. While this might increase computational costs, the trade-off could be worthwhile in scenarios where handling rare tokens is a priority.

5.2 Low-level Vocabularies

In previous sections, we have explored various vocabulary encoding strategies for NMT. In this section, we briefly discuss low-level vocabularies such as byte- and character-level vocabularies, to contextualize a new category proposed by us, called *Quasi-character-level vocabularies*, which aims to combine the benefits of low-level encodings with the efficiency of high-level encodings (Carrión & Casacuberta, 2021, 2022c).

5.2.1 Byte- and Character-Level Vocabularies

Byte-level and character-level vocabularies encode text as sequences of bytes or characters, respectively (Costa-jussà et al., 2017; Vilar et al., 2007). This approach ensures that any word, including rare or unseen ones, can be represented without introducing OOV tokens. The primary advantages of byte- and character-level encodings are:

- **Universal Representation:** They can represent any text input, eliminating the OOV problem inherent in fixed vocabularies.
- **Language Agnostic:** These encodings do not rely on language-specific tokenization rules, making them suitable for multilingual applications.

- **Fine-Grained Compositionality:** By operating at the byte or character level, models can learn to compose words and meanings from the most basic units.

However, the primary drawback is that these methods generate longer input sequences compared to word- or subword-level encodings, which result in an increased computational complexity. Additionally, these methods suffer from a lower information density per token, which leads to a loss of semantic content in the encoded units. As a result, this makes capturing long-term dependencies more challenging due to the inherent limitations of neural models, such as the restricted receptive field of neural networks and issues like vanishing gradients.

5.2.2 Quasi-Character-level Vocabularies

Building upon the previous discussion on the benefits and limitations of low- and high-level vocabularies, this subsection introduces the concept of *Quasi-character-level vocabularies*, a new representation strategy designed to combine the granularity of low-level encodings with the efficiency of subword-level methods. In the following sections, we will provide a formal definition and motivation for this approach, compare their performance against traditional encoding strategies, and evaluate their robustness across a broad range of languages, domains, and model architectures.

Definition and motivation

To address many of the challenges discussed earlier, we introduce a new encoding strategy called *Quasi-character-level vocabularies* (Carrión & Casacuberta, 2021, 2022c). This approach can be defined as:

> *A character-based vocabulary that extends its base character set by incorporating the most frequent character pairs.*

The motivation behind these vocabularies is to preserve the universal representation and language-agnostic properties of character-based encodings while increasing their information density. Hence, by incorporating the most frequent character pairs into the initial character-level vocabulary, we can exponentially reduce the size of the input sequences while having virtually the same capabilities to represent any word.

In practice, this method is somewhat flexible regarding the number of character pairs included in the base set, as we prefer to put focus on optimizing the *information density per token* instead of on the definition. As a result, we aim for a vocabulary size that balances low-level granularity with higher-level efficiency, which, during our experimentation, usually meant finding a vocabulary size in the range of 300 to 2000 tokens, depending on the language and specific requirements.

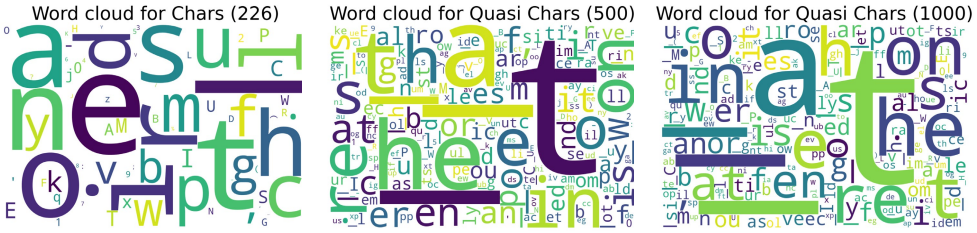


Figure 5.8: Word clouds for character-level and quasi-character-level vocabularies: The figures, from left to right, represent vocabularies of chars (226 tokens), and quasi-chars (500, and 1000 tokens). Notably, the individual tokens in each vocabulary are highly similar, primarily consisting of single characters and a small subset of very frequent character pairs.

Benefits of Quasi-character models

In this section, we discuss the benefits of the proposed quasi-character models, which aim to combine the strengths of low-level encodings with the efficiency of high-level encodings. By extending character-based vocabularies with the most frequent character pairs, quasi-character models provide a balance between flexibility and efficiency that is particularly advantageous in continual learning scenarios for NMT. Consequently, we claim that the primary benefits of quasi-character models are:

1. Retaining the universal representation capabilities of character-based models.
2. Exponential reduction in the number of tokens required to encode sentences when compared to pure character-level vocabularies.
3. Improved learning efficiency.
4. Producing shorter sequences to ease the learning of linguistic patterns.
5. Reducing memory requirements, which is critical for scalability.

We elaborate on each of these benefits in the following paragraphs, highlighting their relevance to continual learning in NMT.

1. Retaining universal representation capabilities

Quasi-character vocabularies inherit the fundamental properties of character-based models. Since they are built upon a base set of characters, they maintain the ability to represent any word, including OOV words, which is crucial in continual learning where new words may appear over time. For example, in a character-based vocabulary, the word ‘‘`hello`’’ is encoded as ‘‘`h + e + l + l + o`”. In a quasi-character vocabulary, frequent character pairs such as ‘‘`he`’’ might be included as tokens, allowing the word to be encoded more efficiently, e.g., ‘‘`he + l + l + o`’’. This maintains the compositionality and flexibility of character-based models while improving efficiency.

In the context of continual learning, the capacity to represent new or rare words without requiring vocabulary updates is highly beneficial. As the model encounters new words over time, the quasi-character encoding ensures that these words can be represented and learned without the need for retraining the embedding layer or dealing with OOV tokens, thus supporting the seamless integration of new vocabulary terms.

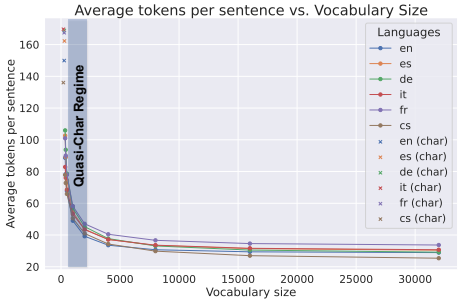
2. Exponential reduction in the number of tokens required

By incorporating frequent character pairs into the vocabulary, quasi-character models can significantly reduce the number of tokens required to encode a particular sentence compared to pure character-level vocabularies, often following a trend where initial gains are exponential relative to pure character encoding as frequent pairs are added. This reduction is important because shorter sequences are easier for models to process and learn from.

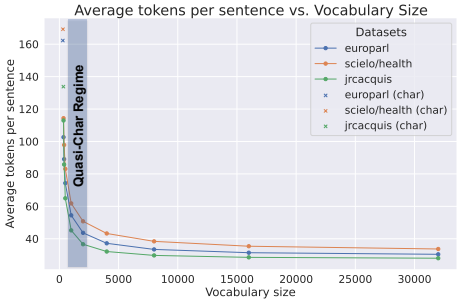
In Figures 5.9a and 5.9b, which replicate Figures 5.2 and 5.3 from Section 5.1.2, we highlight the *quasi-character regime* in blue. This regime utilizes minimal subword-level vocabularies[2] that behave like character-level vocabularies in terms of generalization but, at the same time, benefit from the exponential reduction in sequence lengths of higher-level vocabularies. This balance is particularly beneficial in continual learning, where efficient encoding helps to

[2]It is important to enforce a character-based vocabulary first, and then limit the maximum length of the remaining tokens.

manage the computational resources over long-term training and reduces the risks of performance degradation.



(a) Quasi-character regime for different languages



(b) Quasi-character regime for different domains (datasets)

Figure 5.9: Visualization of the quasi-character regime (blue area), for (a) different languages and (b) different domains (datasets). In both cases, the results exhibit remarkable similarities across different languages and domains.

3. Improved learning efficiency

Quasi-character vocabularies exhibit a high token utilization due to their shorter long-tail distribution compared to high-level vocabularies (see Figure 5.10). This means that all the tokens in the vocabulary are used more frequently (in absolute terms), allowing the model to learn more effective representations for each token.

This is particularly convenient in low-resource scenarios, where the scarcity of training data can hinder the model's ability to learn and generalize efficiently. Thus, by ensuring that each token is represented and used consistently, quasi-character vocabularies enable the learning of more robust embedding representation and the capture of essential linguistic patterns with fewer samples.

Similarly, this is interesting for continual learning too, where models need to adapt to new data continuously, since a high token utilization ratio ensures that the model can efficiently learn new patterns without being hindered by the rarely used tokens.

Consequently, Figure 5.10 compares the token frequency distribution across character-level, quasi-character-level, and standard subword vocabularies. Given the large number of tokens (266 to 15,824), labeling each one along the x-axis

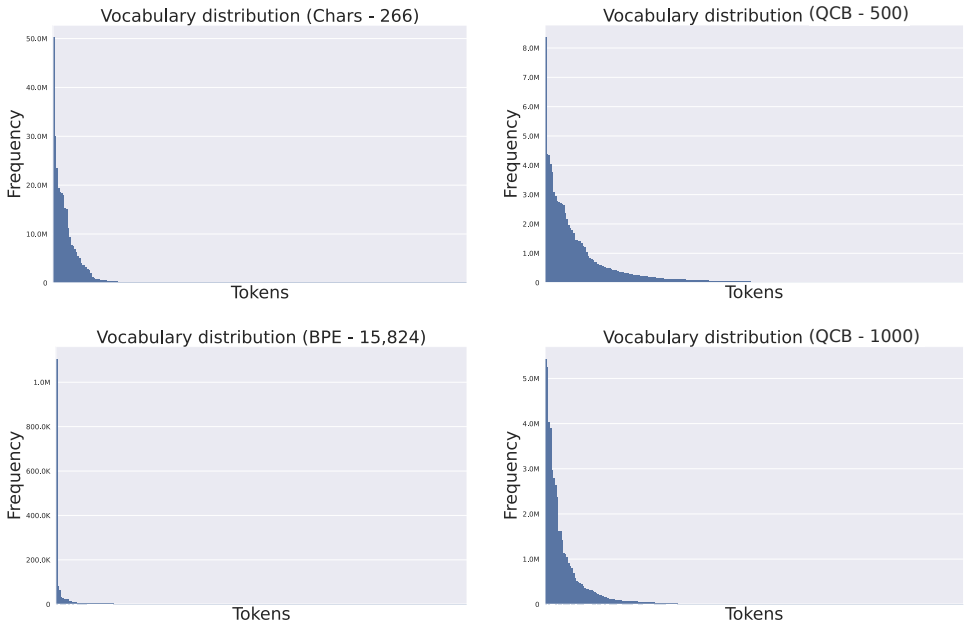


Figure 5.10: Comparison of token frequency distributions across character-, subword- and Quasi-Character-level vocabularies: These figures illustrate the frequency distributions of tokens for various vocabulary encodings, including character-level (266 tokens; top-left), subword-level (16K tokens; bottom-left) and quasi-character-level (500 and 1,000 tokens; top-right and bottom-right), emphasizing their long-tails. The horizontal axis represents individual tokens, while the vertical axis indicates their corresponding frequencies. Due to the large number of tokens per vocabulary, labeling each token on the x-axis was impractical. Notably, character-level and quasi-character-level vocabularies display shorter long-tail distributions that lead to higher token utilization compared to standard subword-level vocabularies (e.g., BPE-16K).

(*Tokens*) was impractical, so token labels have been omitted. As shown, all distributions exhibit a pronounced long-tail behavior. However, both character- and quasi-character-level vocabularies show a more balanced distribution compared to standard subword vocabularies, which enhances the learning efficiency and supports better generalization in low-resource scenarios and continual learning scenarios. This uniform token usage helps to achieve a faster adaptation and reduces the need for extensive retraining when new data is introduced.

4. Facilitating the learning of long-term dependencies

The shorter sequences resulting from using a quasi-character encoding help models capture long-term dependencies more effectively. This is convenient for all neural architectures, but especially for the Transformers models, which have limitations in handling very long sequences due to their quadratic computational complexity with respect to sequence length, thus, reducing the sequence length without losing representational power is critical nowadays.

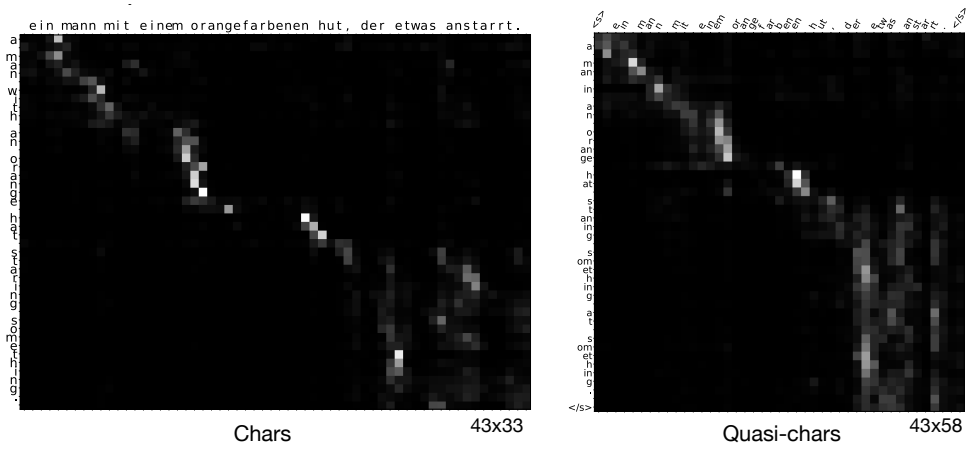


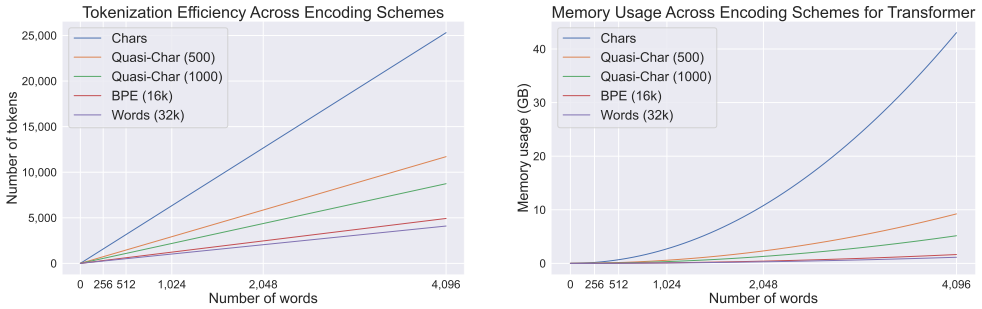
Figure 5.11: Attention heat maps for character- (left) and Quasi-Character-based (QCB) (right) Transformer models: These figures provide an illustrative example of the differences in attention mechanisms between the two encoding strategies. Despite the apparent pattern similarity, the Quasi-Character-based (QCB) model resulted in a faster and better performance, which implies an improved ability in capturing long-term dependencies than the purely character-based models, a trend that was consistently observed across different language pairs and domains.

As shown in Figure 5.11, the attention heat maps from quasi-character models showed patterns similar to those observed in character-based models. However, although the pattern differences were subtle, they are accompanied by an improved performance and a faster learning rate, which underscores the benefits of quasi-character models in capturing long-term dependencies efficiently compared to purely character-based approaches. While these attention patterns are drawn from specific examples, they reflect a broader trend observed across various tasks and language pairs. We expect that this facilitates the model's ability to integrate new knowledge while retaining previously learned information, a key aspect of continual learning, known as the Catastrophic

Forgetting (CF) problem. Therefore, by making it easier to capture long-term dependencies in shorter sequences, we argue that quasi-character models can help to prevent the degradation of performance on previous tasks when new tasks are learned (see Section 6.2).

5. Reducing memory and computational requirements

As shown in Figure 5.12a, the number of tokens increases linearly with the number of words, regardless of the encoding strategies, with the main difference being the slope, which corresponds to the tokens-per-word ratio.



(a) Tokens/Words: Number of tokens as a function of the number of words for different encoding strategies.

(b) Memory/Words: Memory footprint as a function of the number of words for different encoding strategies.

Figure 5.12: Comparison of encoding strategies in terms of token generation and memory usage: The subfigures illustrate the relationship between: (a) the number of tokens produced, and (b) the memory footprint for different encoding methods.
, as a function of the number of words.

However, in standard Transformer architectures, which have a quadratic attention complexity, the memory consumption grows quadratically with respect to the number of tokens.[3] Hence, this quadratic growth in memory usage becomes particularly problematic for low-level encodings, such as byte-level or character-level, due to their higher *tokens-per-word* ratio compared to subword or word-level encodings. It is worth pointing out that even though recent research has introduced Transformer variants with linear-time attention mechanisms to alleviate the quadratic complexity, such as the Reformer (Kitaev et al., 2020) or the Linformer (S. Wang et al., 2020), reducing the sequence length still provides tangible benefits. That is, even with linear-complexity

[3]The number of tokens can be considered similar to the number of words.

attention, fewer tokens translate into less total computation per sequence, making Quasi-Character-based (QCB) encodings advantageous in both standard and linear-time frameworks. Therefore, by reducing the *tokens-per-word* ratio via a QCB encoding, we can significantly reduce the memory footprint of the models while retaining their low-level encoding advantages.

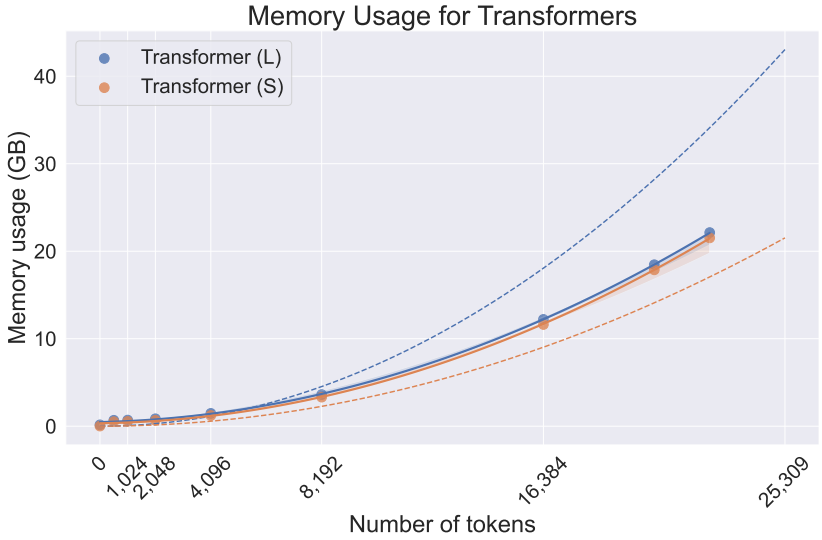


Figure 5.13: Memory footprint analysis of the Transformer architecture: The figure shows the memory usage (in Gigabytes) of a Transformer model as a function of the number of input tokens. The solid lines represent measured data points from a real-world scenario (model loaded on GPU), while dashed lines indicate theoretical approximations, which exclude memory overheads and specific model optimizations, providing a rough confidence interval. The *Transformer (L)*, in blue, corresponds to the standard Transformer with 40M parameters, whereas the *Transformer (S)*, in orange, represents a smaller version with 10M parameters.

This is evident in Figure 5.13, which shows the measured memory consumption of a Transformer model (large and small) as a function of the number of tokens in the input sequence[4]. There, we can see how memory consumption increases quadratically, reaching around 20GBs for input sequences with just 20k tokens. This emphasizes the importance of reducing the number of tokens to efficiently take advantage of our computational resources.

In the context of continual learning, QCB vocabularies provide an efficient and scalable solution for NMT systems. By balancing the flexibility of character-

[4]We assume a similar output length sequence.

based encodings with the efficiency of higher-level encodings, QCB models enable continuous adaptation to new data and dynamic vocabularies without imposing excessive vocabulary limitations or computational costs. This makes them interesting for real-world applications where the universal representation capabilities and optimized resource management are critical to ensure a robust performance over time.

Quasi-characters vs. Other vocabularies

To evaluate the effectiveness of QCB vocabularies compared to other encoding strategies, we conducted an experiment similar to the one described in Section 5.1.3 but including the training runs of the proposed Quasi-Character-based models (see Figure 5.14). Specifically, we trained six QCB models, with two vocabulary sizes (one with 500 tokens and another 1000 tokens) on the same dataset as the previous experiment (Europarl English-Spanish), and again, with three different training volumes: low-resource (50k sentences), mid-resource (100k sentences), and high-resource (200k sentences). We later repeated this experiment for other language pairs and domains with similar results. In the following subsections, we will expand this experimentation to generalize our findings across different languages, domains, and model architectures.

As a result, in Figure 5.14 we present the validation results in terms of BLEU scores obtained during the training of these models. This comparison highlights the performance of different vocabulary encoding strategies across varying training data sizes, with a particular focus on the learning curves and convergence rates rather than solely on the final scores[5]. From this Figure, we confirm that QCB models effectively help to bridge the gap between low-level and high-level encodings, offering a balanced approach that performs well across various resource settings. Furthermore, the analysis of learning curves revealed that QCB models tend to converge faster than byte-level and character-level encodings in low- and mid-resource settings, while maintaining comparable stability in high-resource scenarios.

However, it is worth to mention that our experiments yielded some key observations about these QCB models:

- **Superior performance in low- to mid-resource scenarios:** In low-resource (50k sentences) and mid-resource (100k sentences) scenarios, QCB models outperformed all the other encodings (bytes, char, standard

[5]The scores for the test sets were remarkably similar to those of the validation sets, as both evaluation sets were sampled from the same dataset.

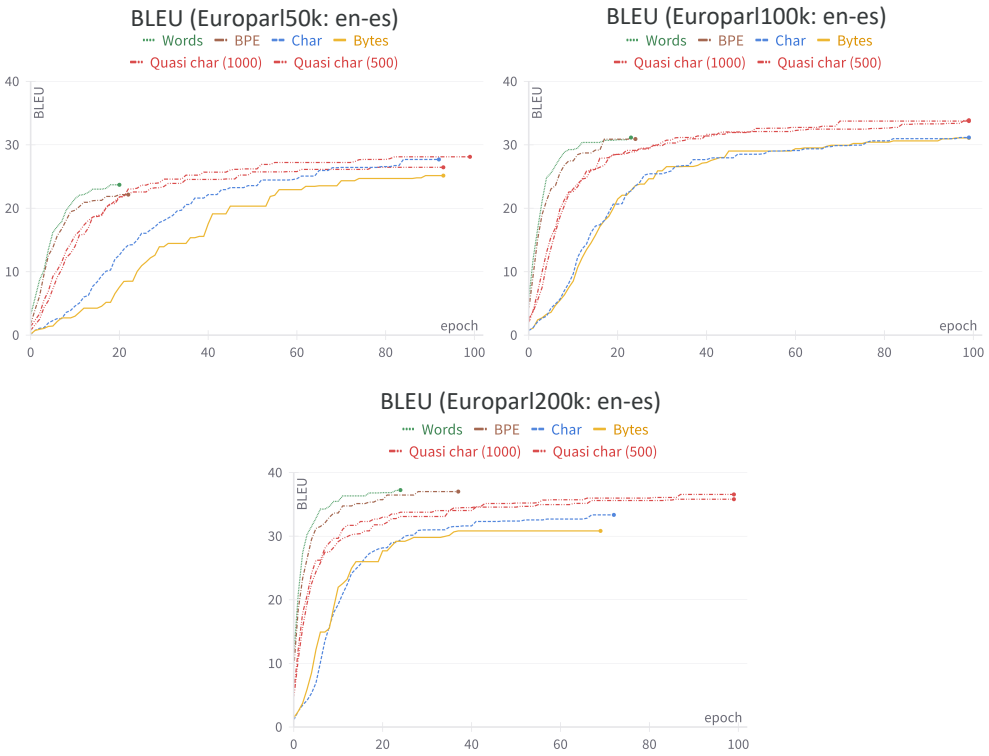


Figure 5.14: Comparison of QCB encodings by training volumes (50k, 100k, 200k sentences): QCB models trained with vocabulary sizes of 500 and 1000 tokens consistently outperform low-level encodings for all training sizes. They also surpass subword vocabularies in low- to mid-resource scenarios, while in high-resource, high-level encodings showed the best performance.

subwords, and words). This shows the effectiveness of QCB vocabularies in settings where training data is scarce.

- **Higher convergence rate than low-level encoding models:** QCB models exhibited faster convergence rates than low-level models (byte and char-based), regardless of the training data volume (low, mid, and high). This validates the idea that shorter sequences, with higher information density per token, allow for more efficient learning.
- **Similar convergence rate to high-level encoding models:** The convergence rates of QCB models were just slightly worse than those of standard subword- and word-level vocabularies. This indicates that QCB

vocabularies can achieve similar training efficiency while retaining the advantages of low-level encodings.

- **Competitive performance in high-resource scenarios:** While high-level encodings (subword and word-level) surpassed QCB models in high-resource scenarios (200k sentences), the performance gap was relatively small. Hence, QCB models remained competitive, highlighting their robustness across different data volumes.

The results suggest that QCB vocabularies offer a flexible and effective encoding strategy for NMT models, especially in settings where data availability is constrained or where adaptability to new or rare words is critical or in a continual learning scenario, where we look for an efficient universal representation. These proposed vocabularies combine the strengths of low-level encodings in handling OOV words while maintaining benefits associated with high-level vocabulary in terms of sequence length and learning efficiency.

Therefore, QCB vocabularies represent a promising approach for continual learning in NMT, where models must adapt to evolving vocabularies and domains without incurring significant computational overhead or sacrificing performance.

Consistency across languages, domains, and model architectures

In this section, we study the consistency of the previous findings about the QCB models across different languages, domains, and neural architectures. We conducted a series of experiments to assess whether the benefits observed with QCB models remain consistent across these variations.

It is important to clarify that any single score reported in this section corresponds to the test set. However, when learning curves are presented, they are derived from the validation set at training time. Both the validation and test sets were randomly sampled from the training set and subsequently excluded from it to ensure unbiased evaluation.

Language consistency

First, we evaluated the performance of QCB models across a diverse set of languages, including both Latin and non-Latin scripts, as well as high-resource and low-resource languages. The languages selected for this study are summarized in Table 5.2.

Table 5.2: Languages used for evaluating the consistency of QCB models: This table summarizes the languages used, categorized by script type (Latin and Non-Latin) and resource availability (High-Resource and Low-Resource).

	Latin Languages	**Non-Latin Languages**
High-Resource	Spanish, German, French, Italian, Portuguese	Russian Chinese*, Arabic*, Korean*, Japanese*
Low-Resource	Czech, Swedish, Danish	Bulgarian, Marathi*, Oriya*

In Figure 5.15, we present a comparison between QCB models and character-based models using box-and-whisker plots across four different language pairs: two low-resource latin languages (i.e., Swedish-English, Danish-English), one low-resource non-latin language (i.e., Bulgarian-English), and one high-resource non-latin language (i.e., Russian-English), all from the Europarl dataset. These plots illustrate the distribution of BLEU scores, capturing medians, interquartile ranges, and confidence intervals at 95%, which highlight the robustness and consistent performance of QCB models compared to their character-based counterparts.

To standardize this experiment, we limited all training data to 100k sentences; thus, the Russian-English pair could also be considered a low-resource sce-

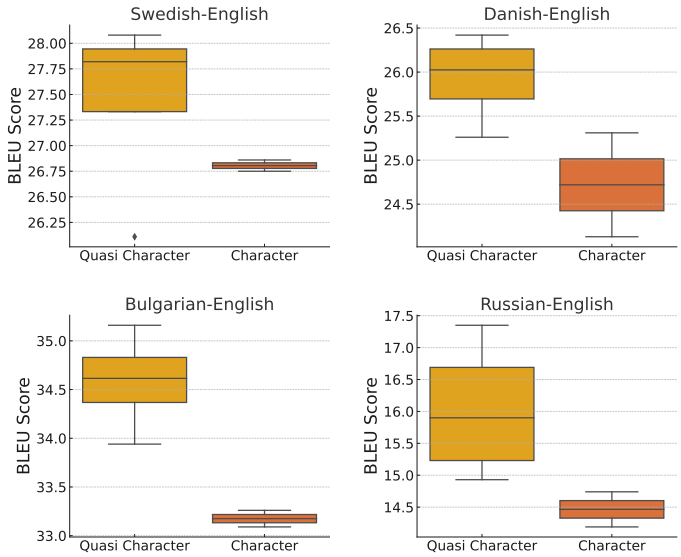


Figure 5.15: Performance comparison of QCB- and Character-based models: The results from evaluating multiple language pairs under controlled low-resource conditions (100k training sentences), show that QCB models consistently outperform the character-based models across all tested language pairs.

nario for this purpose. Consequently, the results indicate that QCB models consistently outperform character-based models across all training runs in the tested low-resource scenarios. In addition to this, we conducted an extension on this low-resource experiment comparing a set of character-based model references against a wide range of standard subword vocabularies (including quasi-character-level vocabularies) using a Unigram encoding (Kudo, 2018), which is a probabilistic variant of BPE, with vocabulary size ranging from 350 to 16,000 tokens. This time, we added byte fallback mechanisms for both character-based and subword-based models to test what is a common strategy nowadays. The results shown in Figure 5.16 reveal a descending trend in performance as the vocabulary size increases, which highlights the fact that QCB models achieved better results in low-resource scenarios than high-level encodings regardless of the language. Similarly, purely character-based models fell between the QCB models and the standard subword models in terms of performance, as expected.

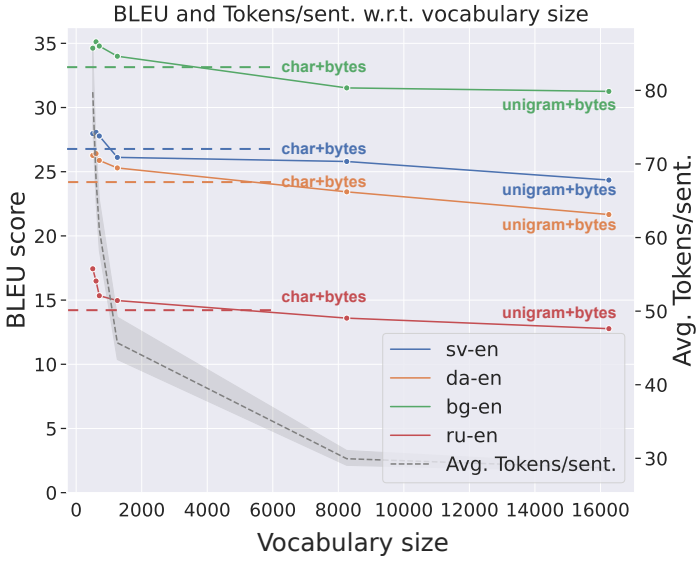


Figure 5.16: Effect of vocabulary size on Low-Resource Machine Translation (LRMT) models: The results from using a Unigram encoding with byte fallback show a decline in performance as vocabulary size increases in low-resource contexts. In contrast, QCB models (solid lines; <2000-4000 vocab size) consistently outperform both character-based models (dashed lines) and standard subword-level models (solid lines; >2000-4000 vocabulary size).

To further validate our previous findings, we extended the comparison between QCB models and standard subword-level models across a few more high-resource language pairs (Spanish-English, German-English, and Czech-English) from Europarl, on different data volume scenarios: low-resource (50k sentences), mid-resource (100k sentences), and high-resource (630k/2M sentences). As in previous experiments (see Section 5.2.2), the results from Figure 5.17 show that QCB models consistently outperformed high-level encodings in low- and mid-resource scenarios, while showing a slightly lower performance in high-resource scenarios when compared to standard subword-level models (Unigram-32k).

These results suggest that the performance trends observed for QCB models hold consistently across a diverse range of languages and scripts.

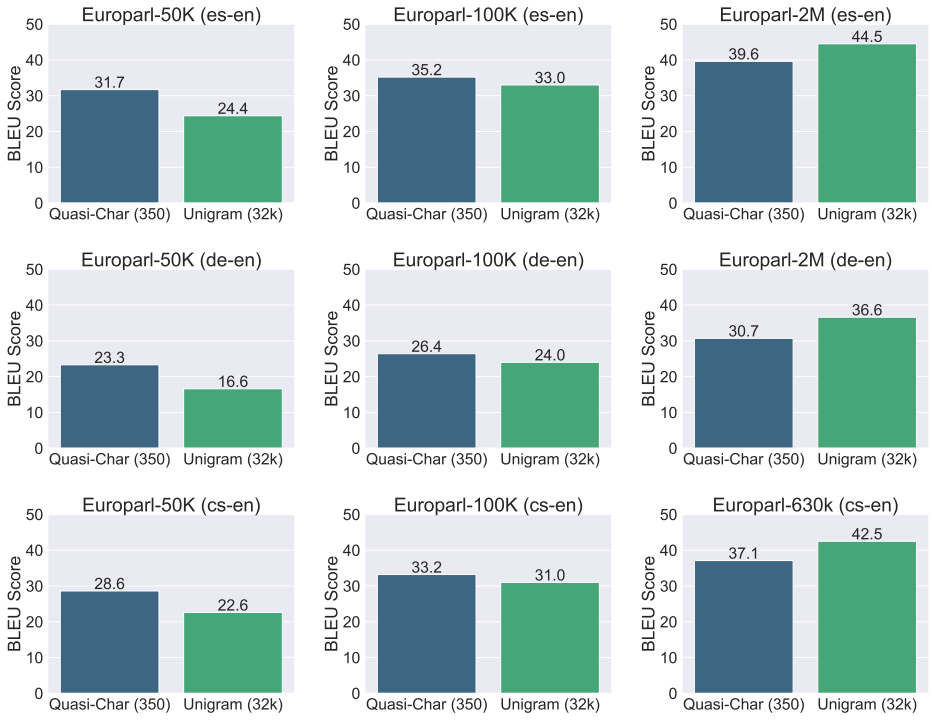


Figure 5.17: Performance comparison in BLEU points between QCB models and standard subword models across various languages and two data volume regimes. In line with previous experiments, the QCB models demonstrated a superior performance in low- and mid-resource scenarios (50–100k sentences) across all languages, and achieved comparable results in high-resource languages (0.6–2M sentences).

Domain consistency

We further investigated the consistency of QCB models across different domains by conducting a series of similar experiments but fixing the language pair (i.e., Spanish-English). The domains included General (CommonCrawl), Scientific (SciELO-Health), News articles (NewsCommentary), and European Proceedings (Europarl).

In Figure 5.18, we present the comparison between QCB models with a vocabulary size of around 350 tokens and standard subword models with a vocabulary size of 32k tokens, evaluated across different data volumes: low-resource (35k

sentences), mid-resource (100-120k sentences), and high-resource (up to 2M sentences).

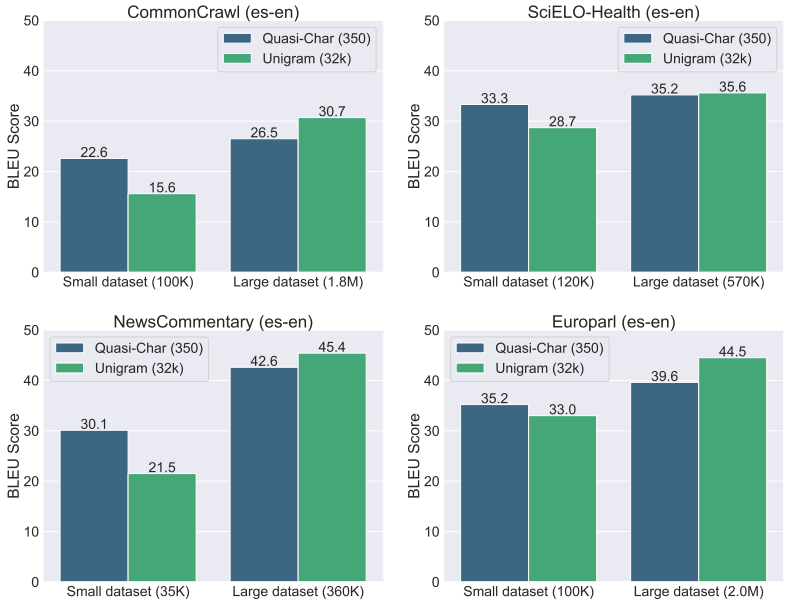


Figure 5.18: Performance comparison in BLEU points between QCB models and standard subword models across various domains and two data volume regimes: The QCB models showed a superior performance in low- and mid-resource scenarios (35–120k sentences) across all domains, and achieved comparable results in high-resource domains (0.5–2M sentences).

Therefore, these results show that the previously presented benefits of QCB models are not limited to specific domains but are broadly applicable across a wide range of scenarios, making them an interesting option for various translation tasks, including continual learning scenarios.

Model architecture consistency

Finally, we further explored the Impact of different vocabulary encodings across several well-established NMT architectures: SimpleRNN (Sutskever et al., 2014), ContextGRU (Cho et al., 2014), AttentionGRU (Bahdanau et al., 2015; Cho et al., 2014), CNN (Gehring et al., 2017), and Transformer (Vaswani et al., 2017)[6]. The goal of this experiment was to evaluate how effectively these architectures could capture long-term dependencies when trained on both low-level encodings (byte, character, and quasi-character) and high-level encodings (standard subword- and word-level vocabularies), as well as to demonstrate the abilities of QCB encodings to ease the learning of these sequences.

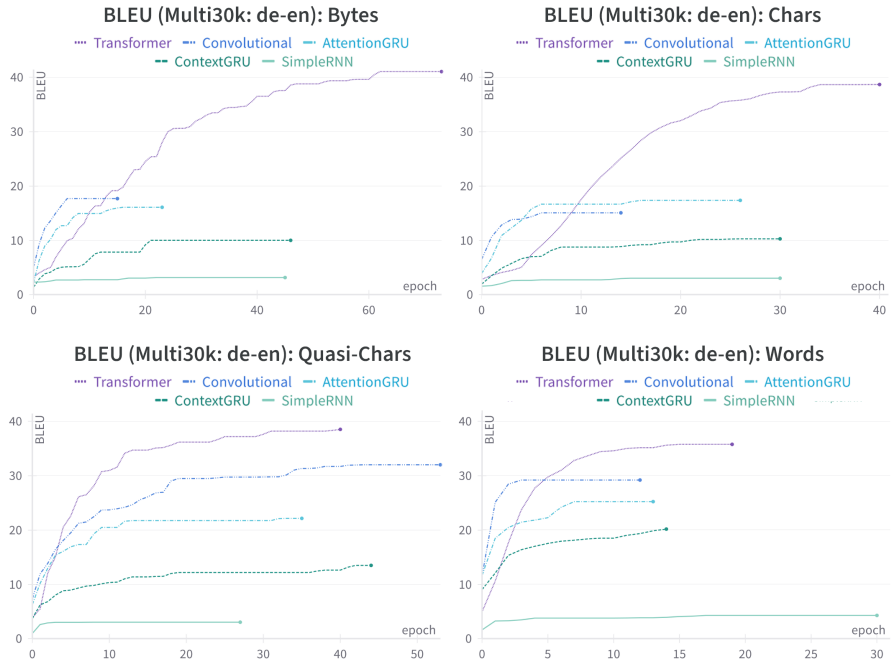


Figure 5.19: Neural architectures comparison (Grouped by encoding): Results on the validation set, measured in BLEU points, for training five different neural architectures (SimpleRNN, ContextGRU, AttentionGRU, CNN, and Transformer) with low-level encodings (byte, character, quasi-character) and high-level encodings (subword, and word). QCB models reduce the performance gap between low-level and high-level encodings by facilitating the learning of long-term dependencies.

[6]We also tested the LSTM architecture in a separate experiment, obtaining comparable results to the GRU architecture.

To do so, all models were configured to have comparable representational capacity (e.g., similar parameter counts, depth, learning rates, batch sizes, etc.), although this provides only an approximate comparison of their theoretical capabilities. As a result, in Figure 5.19 and Figure 5.20, we present the results of this experiment[7], measured in BLEU points, using the validation set during the training of the models. Figure 5.19 organizes the results by encoding type, while Figure 5.20 groups them by architecture type.

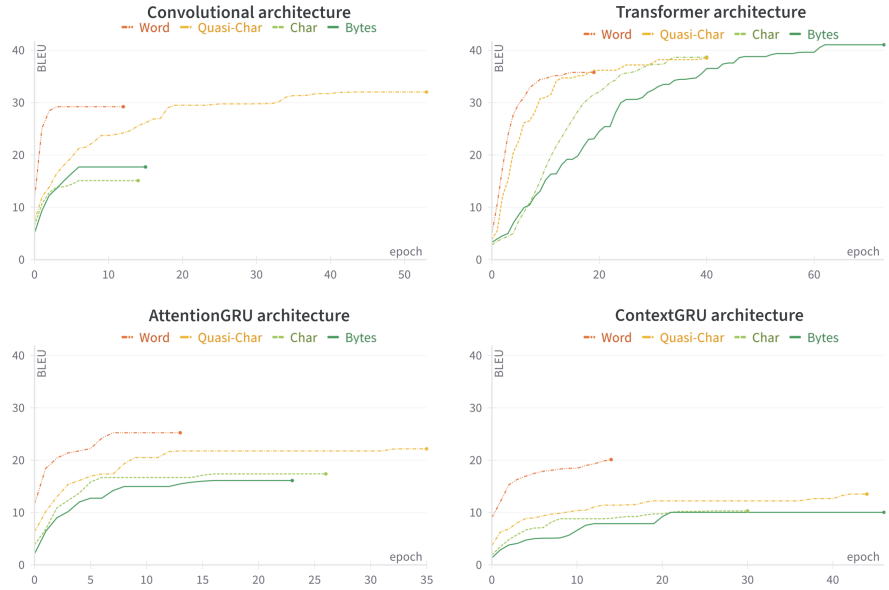


Figure 5.20: Neural architectures comparison (Grouped by architecture): Results on the validation set, measured in BLEU points, for training four different neural architectures (ContextGRU, AttentionGRU, CNN, and Transformer) with low-level encodings (byte, character, quasi-character) and high-level encodings (word). The Transformer consistently outperformed other architectures by modeling long-term dependencies more easily. However, the use of QCB encodings help to improve performance for less effective architectures w.r.t low-level encodings.

The learning curves from the validation set show that the Transformer architecture was the only architecture capable of consistently modeling the long-term dependencies associated with low-level and QCB encodings, a trend that was also observed on the test sets. In contrast, architectures such as Convolutional Neural Networks (CNNs) and Gated Recurrent Units (GRUs)-based

[7]Due to time constraints, we conducted a limited number of experiments using standard subword vocabularies, with 16k tokens, primarily to confirm that these results were consistent with those from the word-based models.

models struggled with byte- and character-level encodings due to the longer sequence lengths and the inherent difficulty of capturing long-term dependencies. However, the use of QCB encodings appeared to significantly improve the performance of these models, narrowing the performance gap between low-level and high-level encodings.

These findings suggest that while the choice of neural architecture significantly affects a model's ability to learn long-term dependencies, QCB encodings can mitigate the challenges typically faced by architectures that used to struggle with low-level encodings like bytes or characters.

In summary, our experiments show that the advantages of QCB models hold consistently across different languages, domains, and neural architectures. Nevertheless, while QCB encodings enhance the ability of all architectures to learn from low-level encodings, we strongly recommend the use of the current State-of-the-Art Transformer architecture, regardless of the encoding employed.

5.3 Incremental Vocabularies

Incremental vocabularies provide a practical and scalable solution for expanding the vocabulary of a pre-trained NMT with new terms without requiring a full model retraining. These methods allow NMT models to stay updated as language evolves over time, ensuring that emergent terms and domain-specific jargon can be seamlessly integrated into their base vocabularies as they arise or in periodic updates.

In the following sections, we will provide a comprehensive definition of incremental vocabularies, highlighting their relevance in addressing the open-vocabulary problem, and discussing strategies for integrating these new terms while preserving their global semantic coherence.

5.3.1 Definition and Importance

In this section, we introduce the concept of *incremental vocabularies*, which can be defined as:

> *A vocabulary that allows the continuous integration of new words into the model without requiring a full retraining.*

Formally, an incremental vocabulary can be represented as an extension to an existing vocabulary $\mathcal{V}$, where a new set of words $\mathcal{V}_{\text{new}}$ is incrementally added:

$$\mathcal{V} \rightarrow \mathcal{V}' = \mathcal{V} \cup \mathcal{V}_{\text{new}} \tag{5.1}$$

This implies that, for each new word $w \in \mathcal{V}_{\text{new}}$, we will have to add an embedding $\mathbf{e}(w)$ that is aligned with the embeddings of the pre-existing words in $\mathcal{V}$. To do so, we learn a mapping function that projects both the new and existing embeddings into a shared latent space. This alignment is achieved by minimizing a divergence loss over a common subset of words, $\mathcal{V}_{\text{common}}$, present in both the old and new vocabularies. By doing so, we ensure that the newly introduced embeddings can be seamlessly integrated into the existing embedding space, preserving the semantic relationships across all words:

$$\mathcal{L} = \sum_{w \in \mathcal{V}_{\text{common}}} \left\| \mathbf{e}_{\text{new}}(w) - \mathbf{e}(w) \right\|^2 \tag{5.2}$$

Therefore, the importance of incremental vocabularies lies in their ability to enable NMT models to adapt to dynamic and ever-evolving environments, such as the emergence of new terms, slang, technical jargon, or acronyms. Hence, by allowing models to incorporate new words seamlessly, we maintain the relevance and accuracy of NMT systems as languages evolve, while significantly reducing the need for expensive and time-consuming retraining. This allows the system to remain effective in zero-shot translation scenarios, where the model encounters previously unseen words.

As an example of this concept, we can look at Figure 5.21, which provides a schematic representation of how an incremental vocabulary can be implemented in practice. The process begins with a base model containing a vocabulary of 32k words, sufficient for most tasks at the time of training. However, as language evolves, new terms emerge (e.g., TikTok, LLM, e/acc,...) that are not represented in the original vocabulary. This causes our model to be outdated, even though the grammatical structures of the languages remain unchanged, making a full retraining or even fine-tuning unnecessary. Instead, a simple vocabulary expansion should be sufficient. To address this, we can apply an incremental update to seamlessly integrate the 2k new words into the model. This approach ensures that the extended model remains relevant and capable of adapting to new linguistic challenges without requiring a costly retraining of the entire system.

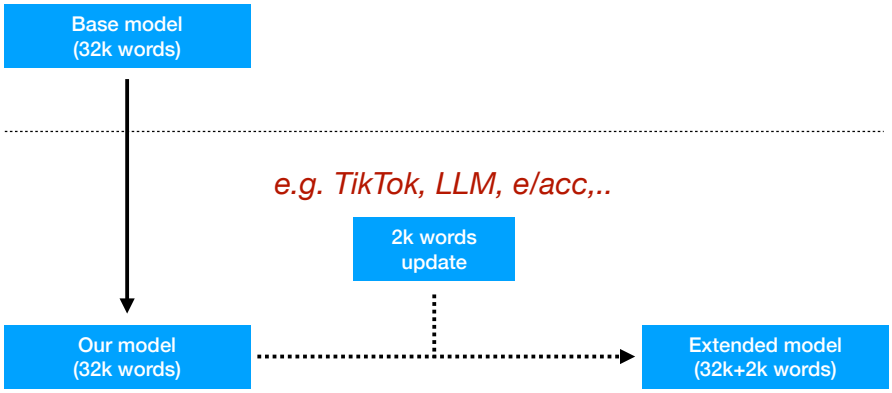


Figure 5.21: Schema of Incremental Vocabulary Expansion: This is a conceptual diagram of a pre-trained model with an initial 32k-word vocabulary that is seamlessly extended with 2k new word embeddings, without requiring a full retraining or fine-tuning. This incremental update strategy ensures that the model remains up-to-date with emerging linguistic terms while leveraging the already learned grammatical structures of the languages involved.

5.3.2 Zero-Shot Translation at the Vocabulary Level

In this section, we explore the definition of zero-shot translation at the vocabulary level, its inherent challenges, and the limitations of current approaches.

Definition

Zero-shot translation at the vocabulary level refers to the model's capacity to generate meaningful representations for, and potentially translate, words that were not encountered during training. This is achieved without the need for additional retraining or fine-tuning, relying instead on compositional semantics, subword decomposition, and alignment techniques (Johnson et al., 2016). Such generalization hinges on flexible vocabulary encoding mechanisms and the model's ability to infer the meaning of unseen terms from their morphological structure and contextual cues (Mullov et al., 2024). By leveraging such methods, the model is expected to handle new terms without explicit exposure during training, thus, enhancing its lifelong capabilities.

Challenges and Solutions

Handling words without direct mappings in the training data poses significant challenges. This occurs because, without explicit examples, the model will struggle, at best, to produce meaningful representations of unseen words. Moreover, the integration of new pre-trained word embeddings into a pre-trained model requires careful consideration, as the new set of embeddings needs to be semantically coherent with the existing ones.

To address these challenges, two solutions have been proposed:

- **Compositional Embeddings**: By exploiting subword information to construct word embeddings compositionally, we can allow the model to infer the embeddings for unseen words based on known subword units. This approach leverages the compositional nature of language to build representations for new words.
- **Embedding Alignment**: By learning a transformation function to align new embeddings with existing ones, we can ensure the preservation of the semantic relationships. This often involves identifying a mapping between the new and existing embedding spaces based on a shared vocabulary.

Limitations of current approaches

Despite these solutions, current approaches have limitations. For instance, compositional methods may not capture the full semantic nuances of new words like TikTok, e/acc, LRMs, LMMs,... (Bojanowski et al., 2016; Devlin et al., 2018; M. Lewis, 2023; Peters et al., 2018). Similarly, embedding alignment techniques often require a substantial amount of overlapping vocabulary between the new and existing embeddings, which may not always be available. Besides, the integration of new embeddings can introduce inconsistencies in the model's latent space, leading to a degradation in performance (Sahin et al., 2017).

5.3.3 Requirements for Unseen Word Embeddings

In order to effectively extend the vocabulary of a pre-trained model with new entries, certain requirements must be met by the unseen word embeddings. In this section, we study the needed requirements to ensure that the new embeddings can be seamlessly integrated into the existing model without disrupting its learned representations (Carrión & Casacuberta, 2022a).

Dimensionality matching

One of the most critical requirements for integrating new word embeddings into an existing model is to ensure *dimensionality matching*. Although the model architecture typically defines a fixed embedding dimensionality (e.g., 512 dimensions), in practice, we often want to integrate new embeddings derived from external sources or different pre-trained models. These external embeddings may have different dimensions, or we may wish to reduce their complexity by projecting them into a lower-dimensional space that matches the original model. Thus, even though the final embedding dimensionality that the model consumes is indeed fixed, the challenge is in matching any newly introduced embeddings, which might come from sources with different dimensionalities, with that fixed and common dimension. This matching is a first step so that the model can seamlessly consume and process the new embeddings as it does for the original ones.

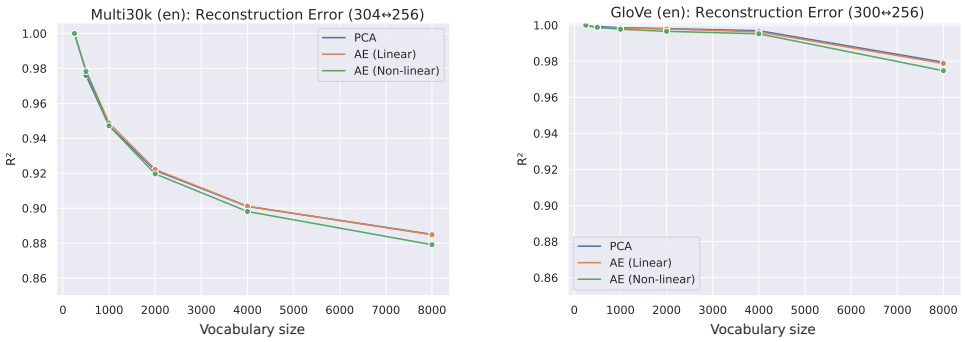
In practice, this issue can be addressed through various projection techniques that map the new embeddings into the same latent space as the original ones. The two most common methods are linear projections (e.g., Principal Component Analysis (PCA)) and non-linear methods (e.g., Autoencoders). Since each of these approaches has its own trade-offs in terms of reconstruction error and computational complexity, we decided to compare the reconstruction errors of these methods. Specifically, we compressed the embedding vectors into a lower-dimensional space and then expanded them back to match the original dimension to compare the reconstruction error[8] w.r.t the original vectors. We repeated this experiment over an incremental set of word embeddings, from 250 embeddings to 8000 embeddings, sorted by frequency. We did this to study how the quality of these representations, using the word frequency as a proxy, affected these compressions.

For our experiments, we used two main sources of embeddings: (1) low-quality embeddings obtained from a small Transformer model trained on the Multi30K dataset (Elliott et al., 2016), a relatively small and limited corpus of about 30k image-caption pairs, and (2) high-quality embeddings from the GloVe vectors (Pennington et al., 2014), that derive from modeling the co-occurrence probabilities through a matrix factorization process on a large dataset (i.e., Wikipedia and Common Crawl). Additionally, we also tested larger embedding sets derived from the Europarl dataset (German-English, with 2 million sen-

[8]We used the coefficient of determination, $R^2 = 1 - \frac{\sum_i \|\mathbf{x}_i - \hat{\mathbf{x}}_i\|^2}{\sum_i \|\mathbf{x}_i - \bar{\mathbf{x}}\|^2}$, which measures how well the reconstructed vectors $\hat{\mathbf{x}}_i$ approximate the original vectors $\mathbf{x}_i$. A value of 1 indicates perfect reconstruction.

tences) to confirm that our observations generalize to other datasets. In all cases, we projected embeddings into a target dimension (i.e., 256) and then re-projected them back to their original dimensionality, measuring how well each dimensionality reduction technique preserved the original embedding space (reconstruction error).

In Figure 5.22, we present the reconstruction errors for both, the low-quality (Multi30K) and high-quality GloVe embeddings, under linear PCA and non-linear (autoencoders) dimensionality reduction methods, as well as a linear autoencoder. This evaluation allows us to compare how different approaches scale with increasing vocabulary sizes and varying embedding quality.



(a) Low-quality embeddings.

(b) High-quality embeddings.

Figure 5.22: Reconstruction Errors of Embeddings: On the left, the reconstruction errors (R^2) for low-quality embeddings, illustrating larger errors and higher variance. On the right, the reconstruction errors for high-quality embeddings, showing smaller errors and more consistency. These comparisons highlight (1) the superiority of high-quality embeddings in preserving semantic information and (2) the adequacy of linear methods for capturing relationships in embedding spaces. The higher the R^2, the better the reconstruction.

From these figures, we observe that higher-quality embeddings (i.e., GloVe) exhibit smaller reconstruction errors than lower-quality embeddings (Multi30k). This indicates that the original quality of the embeddings significantly affects the effectiveness of the dimensionality reduction. Additionally, non-linear methods, such as autoencoders, do not provide significant advantages over linear methods like PCA in this context, likely because the embeddings primarily capture linear relationships among words, making linear methods sufficient for dimensionality matching.

Moreover, as the vocabulary size increases, the reconstruction errors tend to rise, regardless of embedding quality. This can be attributed to the long-

tail distribution of word frequencies in natural language, by which: *as the vocabulary expands, less frequent words are included.* This means that the quality of their embedding representations will be worse due to the fewer training samples from which to learn, and since noise cannot be compressed effectively, lossy compression methods like PCA or autoencoders will result in higher reconstruction errors for these embeddings as the vocabulary grows.

Finally, we repeated this experiment using larger embedding dimensions (e.g., 512) and larger datasets (e.g., Europarl-2M de-en), with similar results. Consequently, we conclude that linear projection methods are sufficient for dimensionality matching when integrating new embeddings into existing models. However, the quality of the embeddings remains a critical factor in the success of these techniques. That said, dimensionality matching alone is not sufficient for integrating new words into existing models, as we will explore in the following sections.

Embedding quality

As discussed in the previous section, the quality of word embeddings plays a critical role in ensuring that new embeddings can be effectively integrated into a model without significant issues.

To study this phenomenon in more detail, we trained multiple sets of embeddings using progressively larger amounts of training data (30k, 100k, and 2M sentences), as a proxy of the final quality. We then projected these embeddings into a two-dimensional space using t-SNE (van der Maaten & Hinton, 2008) to visualize and gain intuition about the relationships formed between words. Consequently, as shown in Figure 5.23, as the training data increased, the embeddings trained on these large datasets formed tighter and more distinct clusters. That is, words showed higher similarity within groups of related words (better intra-cluster convergence) as well as a stronger differentiation between unrelated concepts (better inter-cluster divergence). In contrast, the embeddings trained on the smaller datasets formed diffuse clusters with poor separation. However, it was not until we used larger amounts of training data (i.e., 2 million sentences) that we began to observe clear clusters of similar words.

Following this experiment, we hypothesized that as we increased the amount of training data, the embeddings would not only improve in quality but would also begin to converge to the same regions in the latent space, regardless of the dataset used. To test this theory, we trained another set of high-

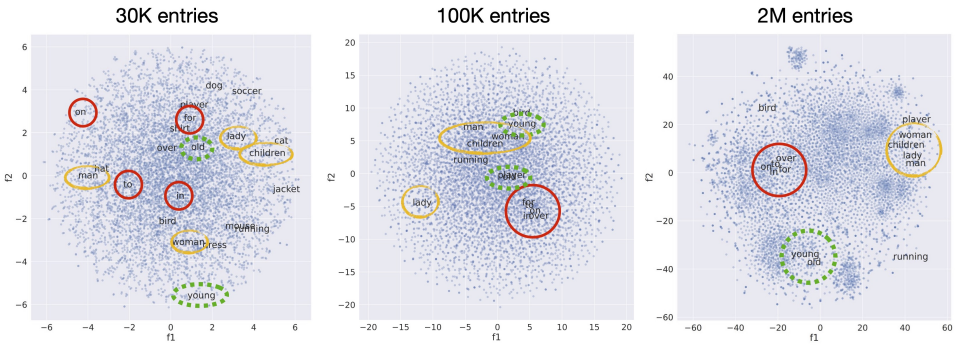


Figure 5.23: Visualization of embedding quality improvement with increased training data: As the training data increases, the embeddings form clearer clusters, indicating better intra-cluster convergence and inter-cluster divergence. The F1 and F2 axes represent the two dimensions of the reduced feature space produced by the t-SNE projection, where relative distances between points reflect the similarity of their high-dimensional embeddings.

quality embeddings on the Europarl-2M dataset. However, we ended up with a completely different set of high-quality embeddings with distinct cluster formations.

To further investigate the formation of these different clusters among high-quality embeddings, we repeated the projection experiment using other sets of very high-quality embeddings such as FastText (Bojanowski et al., 2016) and GloVe (see Figure 5.24). Yet again, we observed that the distances between clusters and the regions of the space where they reside did not appear to converge despite the high quality of the embeddings. Later, we projected these embeddings multiple times and measured their average distance in the high-dimensional space, achieving similar results.

Therefore, these observations seem to suggest that while increasing the amount of training data improves the quality of word embeddings, the specific formation and location of clusters in the latent space are influenced by factors beyond just data volume, such as the stochasticity inherent in the training process, random initializations, variations in training procedures, etc. Hence, despite of their high-quality, different embeddings do not necessarily have to converge to the same regions in the latent space. This finding has important implications for the integration of new embeddings into existing models, indicating that embedding quality alone may not guarantee seamless integration without careful alignment or adaptation.

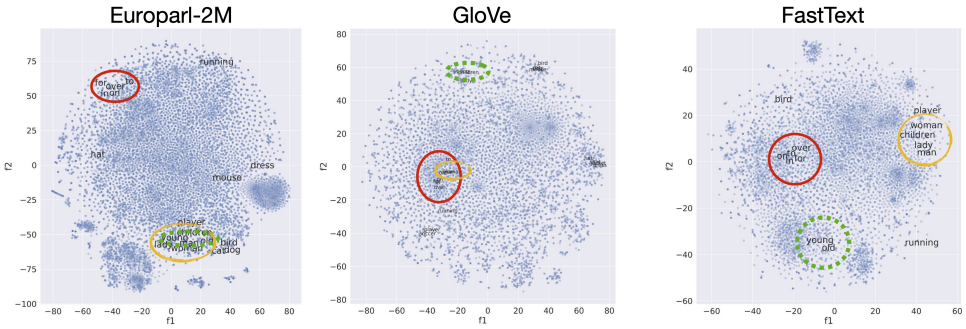


Figure 5.24: Visualization of embedding convergence across different datasets: Despite high-quality embeddings, clusters do not appear to converge to the same regions in the latent space. Again, the axes represent the dimensions of the reduced feature space produced by the t-SNE projection.

Embedding alignment

From the previous results, it became clear that despite the high quality of the embeddings, the new embeddings do not necessarily align with the existing ones. This is because, although they might achieve semantic coherence within their own latent space, this semantic coherence is not universal. Thus, when these semantically coherent embeddings are combined with a different set of semantically coherent embedding sets, they will likely produce a semantically incoherent new set of embeddings.

In Figure 5.25, we illustrate this explanation, where although two sets of embeddings (represented by dashed green lines, or the red lines) may contain semantically similar words (e.g., cat, dog, mouse,...), these embedding sets might be located in different regions of the same latent space. Therefore, this poor inter-cluster alignment might likely lead to a semantic incoherence state when both embedding sets are integrated into the model.

To address this semantic incoherence between different high-quality embedding sets, we explored two solutions. First, we used a compositional approach, which exploits subword information to construct word embeddings compositionally. Specifically, we constructed the base vocabulary using FastText (Bojanowski et al., 2016), allowing us to generate new embeddings for unseen words that are automatically aligned with the existing ones. Second, we tried alignment via linear mapping, which requires the learning of a linear transformation to be able to align new embeddings with the existing ones. This approach first requires a common set of words between the pre-trained and new embeddings,

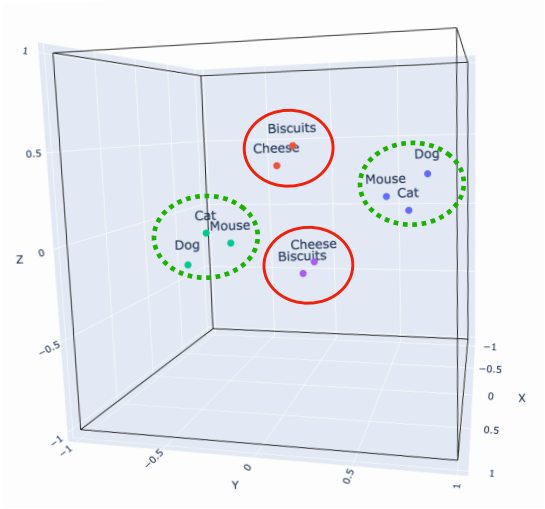


Figure 5.25: Illustration of semantic incoherence between embedding sets: Even though two embedding clusters may contain the same words and exhibit a strong internal semantic coherence, if these clusters originate from different embedding sets, they might also occupy distinct regions within the latent space. Consequently, merging such clusters into a shared latent space without a proper alignment can result in a semantically incoherent embedding set. The X, Y, and Z axes represent the three dimensions of the reduced feature space obtained through the t-SNE projection.

and then, we can use these common words to learn the mapping function that will align the new word embeddings with the existing ones. However, the main problem with this second approach is that it relies heavily on the availability of a sufficiently large set of shared words between the pre-trained and new embeddings, so if the overlap between the two vocabularies is small, the learned mapping may not be reliable.

For the compositional approach, we began by training a base model on the Europarl dataset (German-English), with a decouple compositional vocabulary (FastText-based). However, although compositional vocabularies can generate semantically coherent embeddings for unseen words, each base model was restricted to use only the embeddings included in its initial vocabulary (e.g., from 250 to 500 up to 8000 words). This restriction was done as a measure to ensure that the internal representations connecting the source and target embeddings were derived exclusively from their base vocabularies. This was important because, since we intended to expand these vocabularies in a zero-

shot manner later, it was essential to avoid any information leakage that might give this approach an unfair advantage.

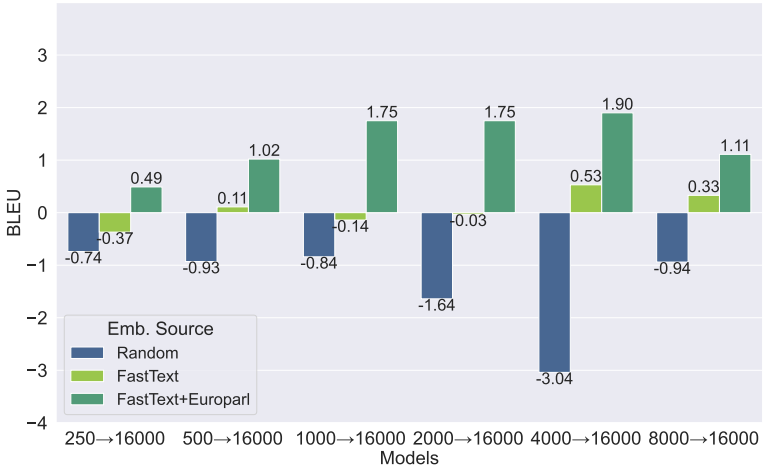


Figure 5.26: Relative improvements in translation quality through vocabulary expansion: The performance gains are compared relative to the base model (only trained with the base vocabulary).

Figure 5.26 shows the relative improvements of this zero-shot vocabulary expansion strategy in relation to the base model. Here, we expanded the vocabulary of our base model from its initial base size (ranging from 250 to 8000 words) to 16k words using three embedding strategies: 1) Random embeddings as a control set; 2) Compositional and semantically coherent embeddings from FastText; and 3), Compositional embeddings (the previous ones) with a minimal fine-tuning for adjusting the internal representations of the model.

From these experiments, we observed that when we used random embeddings (the control set), we lost a significant amount of performance in all vocabulary expansions (as expected). Next, when we used a pure compositional approach to expand the smaller base vocabularies (250-2000 tokens) to the target size (16k tokens), we also found no improvement in performance. We argue that this was because the highly limited amount of base embeddings prevented the model from learning the sufficiently rich internal representations needed to extrapolate to the new embeddings. In contrast, when the base vocabulary was larger (4000 or 8000 words), the zero-shot vocabulary expansion strategy provided small but consistent performance gains. Finally, after applying a very minimal fine-tuning for tuning these internal representations after the

vocabulary expansion, the performance of these models improved significantly with regard to the base model (see Figure 5.27).

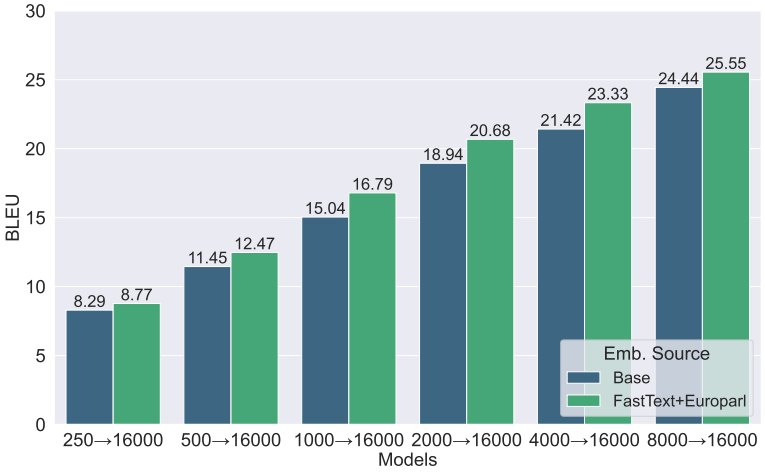


Figure 5.27: Absolute improvements in translation quality through vocabulary expansion: The performance in BLEU points of the corresponding base model (only trained with the base vocabulary), and the same model after the vocabulary expansion and a very minimal fine-tuning process.

We argue that the modest performance improvements from zero-shot vocabulary expansion stem from the fact that word embeddings are only a small and relatively straightforward component of a machine translation system. That is, the model still needs to learn the complex patterns and relationships between source and target languages, so even with a set of two perfect word embeddings for each language, the NMT model still needs to learn how to accurately map these embeddings to produce an effective translation. Therefore, without a minor fine-tuning, the model will likely struggle to fully integrate new words (despite the use of universal word embeddings to mitigate this problem). Additionally, it is highly convenient to start with a sufficiently large base vocabulary to facilitate zero-shot vocabulary expansion and thus, minimize the fine-tuning needed to adjust the internal representations that produce the mapping between new words and their corresponding target pairs.

For the linear mapping approach, we also trained a set of base models on the Europarl dataset (German-English) with a set of fixed incremental vocabularies. We then used the compositional embeddings (FastText) to expand these base vocabularies to 16k words. Since each training run produces a different set of

vocabulary embeddings (semantically incoherent relative to one another), we had to learn a unique mapping function for each run. After this, our results were remarkably similar to those from our previous FastText approach (see Figure 5.26). That is, we did not observe any benefits for base vocabularies under 2000 tokens and, then, modest improvements above that threshold. However, as in our previous experiment, after performing a minimal fine-tuning to adjust the internal representation of the model, the performance results ended up becoming very similar to those from fine-tuning the models with the compositional vocabulary expansion.

In summary, embedding alignment is a crucial step for integrating new words into a pre-trained NMT model. Although the compositional embedding approach offers a more convenient way to expand these vocabularies without additional fine-tuning, both the compositional and linear mapping methods lead to only modest performance gains. This suggests that the model's internal representations play a much more important role than we initially thought, as they capture the complex patterns and contextual relationships needed for an effective translation, overshadowing the impact of even high-quality embeddings. Consequently, the model must adapt its internal representations to fully integrate new words and map them accurately to their corresponding targets. Therefore, starting with a sufficiently large base vocabulary in a semantically coherent framework can significantly reduce the fine-tuning required for vocabulary expansion up to a few sentences, ensuring robust performance and the seamless integration of new embeddings.

5.3.4 Discussion and Implications

Our exploration of incremental vocabularies highlights both their potential and the challenges in integrating new words into pre-trained NMT models without full retraining. We found that while naively matching the dimensionality of embeddings is necessary, it is not sufficient. The quality of the embeddings is also very important to produce an effective projection. However, embedding alignment is the major challenge that these vocabularies face.

Compositional embeddings offered a practical approach to the embedding alignment problem. However, our experiments revealed that vocabulary expansion without fine-tuning (the strictly zero-shot scenario) provides only modest improvements, unless the model undergoes a minor fine-tuning to adjust the internal representations that map the source and the target values. This suggests that while incremental vocabularies are a promising step toward lifelong

NMT, strategies like universal embedding and adaptive fine-tuning are essential to fully unlock their potential.

5.4 Continual Vocabularies

The term of *continual vocabulary* is closely related to the *incremental vocabulary* concept, as both deal with the challenge of integrating new word tokens into the model's vocabulary (Biesialska et al., 2020; Ke & Liu, 2023). However, while incremental vocabulary approaches typically try to expand the vocabulary in discrete steps and often require some form of fine-tuning or embedding alignment, continual vocabulary methods tend to focus on offering seamless integration of new words or terms without the need for retraining or fine-tuning (Cao et al., 2021; Garcia et al., 2021). This allows the system to evolve alongside expanding linguistic data, ensuring adaptability to both domain-specific terminology and emerging language trends. Consequently, continual vocabularies try to solve a core challenge in NMT, the open-vocabulary problem, by enabling models with the ability to remain effective over time as they encounter previously unseen words or symbols. In addition to this, continual vocabularies are designed to maintain a balance between performance and flexibility, leveraging compositional embeddings to ensure that new tokens are effectively being incorporated without compromising the overall performance (Carrión & Casacuberta, 2023b). As a result, this section explores the continual vocabulary problem, focusing on how the domain of a vocabulary affects the performance of a model and how these vocabularies can be easily implemented in practice for NMT systems.

5.4.1 Impact of Vocabulary Domain on Performance

In our initial exploration of the Continual Learning problem in NMT, we decided to investigate how the domain of a vocabulary would affect the performance of an NMT model (Carrión & Casacuberta, 2023b). We hypothesized that while matching the training domain with the test domain is essential for achieving a good test performance, aligning the model's vocabulary domain with the test domain is also highly important and often overlooked.

To test this hypothesis, we started by conducting an exploratory analysis using the SciELO dataset, which contains domain-specific corpora in the Health and Biological sciences. First, we trained a set of separated NMT models (Transformers) on three different domains (i.e., *Health*, *Biological*, and a *Merged* domain combining both) from the SciELO dataset. For each domain, we

constructed a domain-specific vocabulary using BPE. Then, we evaluated each of these models on the test sets (one per domain) to establish the performance baselines of this experiment. Also, to mitigate potential biases arising from asymmetric volumes of training data across domains, we limited each domain's training set to 100k sentences, ensuring a comparable amount of total tokens.

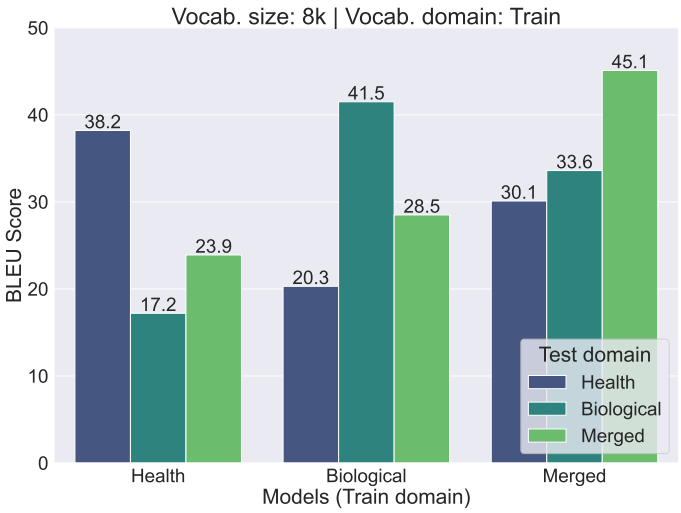


Figure 5.28: Baseline comparison of models trained and tested on three domains: Each model utilized a domain-specific vocabulary generated via BPE from the training set, which was limited to 100k sentences to ensure balanced data volumes. From these results, we set the baselines for studying the impact of both the training domain and the vocabulary domain on translation performance (see Figure 5.30).

As shown in Figure 5.28, each model achieved the highest performance on the test set that corresponded with its training domain, as expected, given the well-established fact that models perform better on data that is similar to their training set. However, to study this more closely, we decided to analyze the overlap between the different domain-specific vocabularies, using the Intersection Over Union (IoU) metric, in order to quantify the degree of vocabulary overlap between domains.

This can be seen in Figure 5.29, where the vocabulary overlap across domains shows a strong correlation between the performance results observed in Figure 5.28, hinting a much more important dependency than previously expected between the alignment of the model's vocabulary with the test set and the model's performance on that test set, regardless of the training domain.

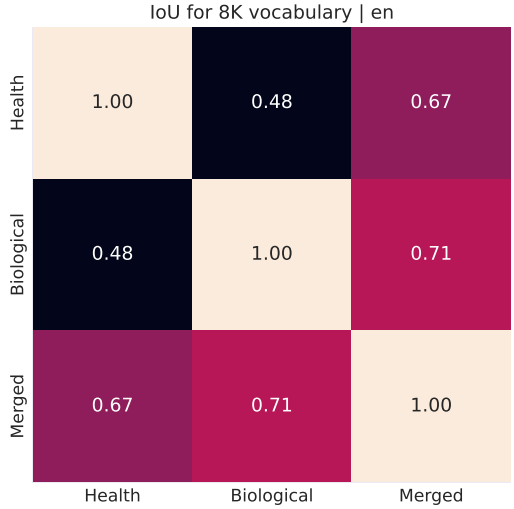


Figure 5.29: Overlap between domain-specific vocabularies: This figure shows the overlap between three domain-specific vocabularies, measured by the IoU metric. This highlights the degree of shared vocabulary across domains and suggests a strong correlation between the model's performance and the alignment of the model's vocabulary with the test domain, regardless of the training domain.

Building on this observation, we decided to extend the previous experiment to test multiple vocabularies of different sizes (i.e., 8k, 16k, and 32k) across the varying domains (i.e., Health, Biological, and Merged). Hence, we trained the models on each domain-specific training set using all these vocabularies, and then, we evaluated their model performance across the different test domains.

As shown in Figure 5.30, the best performance was consistently achieved when the test domain matched the vocabulary's domain, regardless of the training domain. This indicates that aligning the vocabulary domain with the test domain plays a crucial role in improving the translation quality. Additionally, we observed that larger vocabularies do not necessarily result in a better performance. Instead, their effectiveness appears to depend heavily on the volume of the training data, as discussed in previous sections (see Sections 5.1.3). Besides, we speculate this could be attributed to three main factors: (1) Smaller or low-level vocabularies (e.g., bytes, characters, quasi-chars,...) tend to perform better in low- to medium-resource scenarios (Cherry et al., 2018); (2) Each dataset appears to have an optimal vocabulary size, typically forming an

inverted U-shape (Gowda & May, 2020); and (3) Larger vocabularies generally require more training data to achieve optimal results (Sennrich & Zhang, 2019).

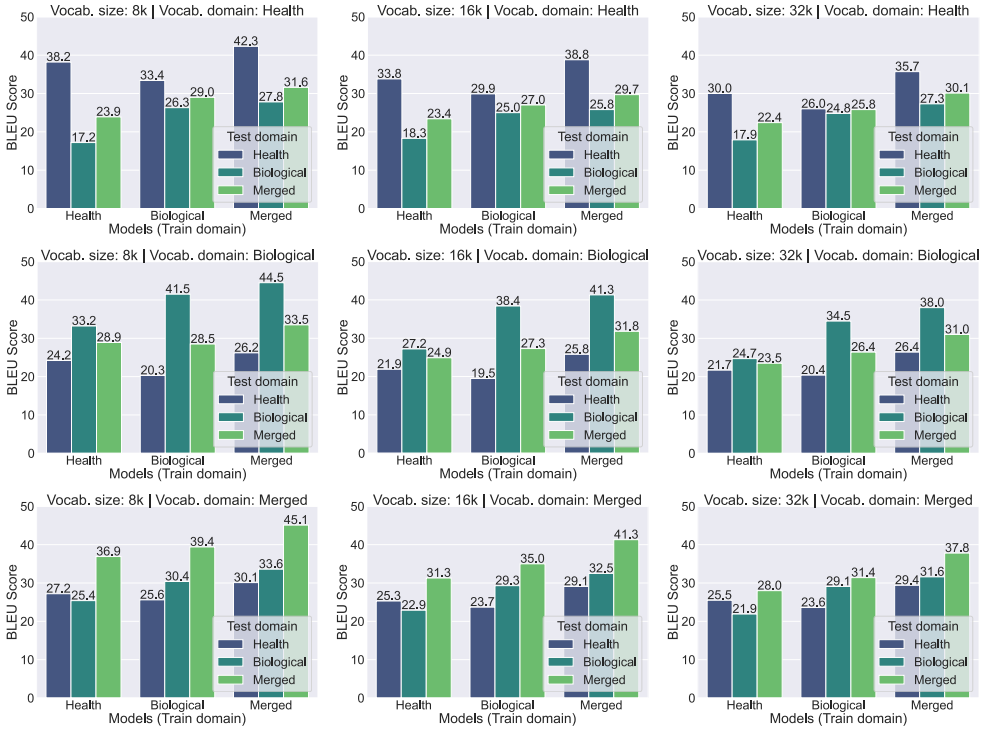


Figure 5.30: Comparison of models trained and tested across three domains as a function of vocabulary size and domain alignment: Each model utilized a domain-specific vocabulary generated via BPE on one of the three training sets (domains), each one limited to 100k sentences to ensure balanced data volumes. The results suggest that aligning the model's vocabulary domain with the test domain is significantly more impactful than previously expected, even overshadowing the contribution of the training domain in this case.

One of the key insights from these results is that, although a general-purpose vocabulary with a good cross-domain generalization is certainly desirable for continual and lifelong NMT tasks, relying on a domain-specific vocabulary that closely matches the test domain is what led the best performance results during our experimentation, to the point of even surpassing those cases where the training domain and vocabulary domain were aligned (provided there is sufficient data). This highlights the importance of aligning the model's vocabulary domain with the test set to potentially maximize translation quality,

an effect particularly pronounced in our experiments with fixed-domain scenarios such as Health sciences or legal applications. Moreover, while larger vocabularies can offer advantages in certain contexts, our experiments suggest that smaller, more specialized vocabularies can be equally (if not more) effective when these are properly aligned with the target domain. Consequently, ensuring that the model's vocabulary is correctly aligned to the application domain is essential for optimizing performance in lifelong NMT systems.

5.4.2 Continual Vocabularies with Compositional Embeddings

To address the continual vocabulary problem in NMT, we proposed a strategy based on a universal and domain-agnostic compositional vocabulary that can be extended over time for new semantics (Carrión & Casacuberta, 2023b). This approach leverages the known subword information, such as character n-grams, to construct new word embeddings compositionally, rather than relying on a static vocabulary fixed at training time. This enables the model to handle a wide range of previously unseen words by inferring their meaning from its known subword components.

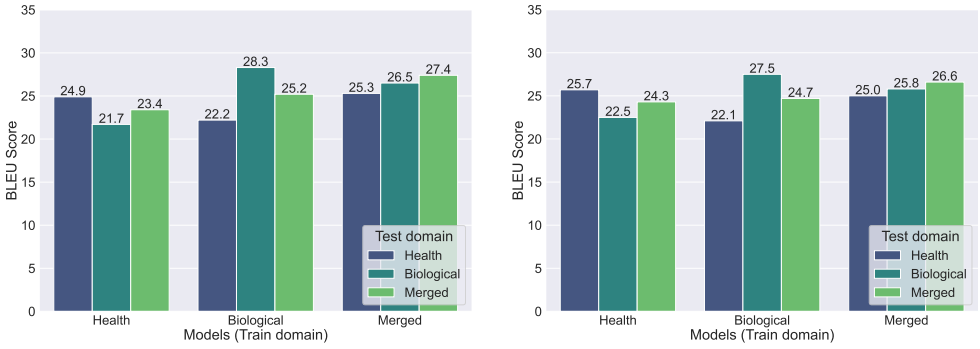
To do so, we first need a base set of vector embeddings (learned from a large and diverse corpus) that supports this compositional approach, such as FastText (Bojanowski et al., 2016). Then, when the model encounters a new term, it constructs its embedding vector by aggregating the embeddings of its constituent subwords (Bojanowski et al., 2016). That is, if the new word can be understood in terms of these known subwords (e.g., *microbiology* can be inferred from *micro* and *biology*), the model will be able to obtain a meaningful representation without any additional training.

This compositional nature ensures that most new tokens can be integrated seamlessly into our model. Because of this, the main advantage of this method is that it typically does not require an explicit vocabulary expansion or fine-tuning process unless entirely novel concepts appear that cannot be inferred compositionally from the existing subwords. For example, this would be the case for new brand names or acronyms (e.g., TikTok, LLM, or e/acc) that may introduce new semantics or semantics not captured by any combination of existing subwords. In such cases, we have to follow an incremental strategy as the one discussed in Section 5.3.

Domain-specific vs. Continual vocabularies

To evaluate the effectiveness of our continual vocabulary approach, we conducted an experiment similar to the one described in Section 5.4.1, with the same models and datasets. However, instead of using domain-specific vocabularies, we employed a continual vocabulary constructed using the FastText library. This library allows the efficient learning of word representations that later, can be generalized to unseen words through compositional embeddings. Specifically, the embeddings were learned from the Common Crawl and Wikipedia datasets.

Accordingly, we trained a few NMT models on the previous SciELO datasets (domains), using this continual vocabulary approach (decoupled), and then, compared their performance with the NMT models that were trained using the domain-specific vocabularies (in-domain vocabularies and tightly coupled with the model).



(a) Domain-specific vocabularies.

(b) Continual vocabularies.

Figure 5.31: Performance comparison of NMT models using domain-specific vocabularies and continual vocabularies across multiple domains (Health, Biological, Merged): The continual vocabulary approach demonstrated a comparable performance to the domain-specific one. These results highlight the robustness of the continual vocabulary approach, which does not compromise the translation quality, while it enables the handling of previously unseen words.

As shown in Figures 5.31a and 5.31b, we can see that the performance differences between the domain-specific vocabularies and our continual vocabulary were minimal across the different domains (test sets). In the health domain, we observed a slight performance improvement of approximately +0.8 to +0.9 BLEU points when using our continual vocabulary approach. In contrast, for the Biological and Merged domains, there was a marginal decrease in performance (-0.1 to -0.9 BLEU points), particularly in cases where the vocabulary domain

matched the training domain. This slight decrease could be attributed to the domain-specific vocabularies being better tailored to the specialized terminology used in those test domains, as they give more precise representations for those domain-specific words (see Section 5.4.1).

However, although these performance differences may appear small at first glance, our results demonstrate that our continual vocabulary approach does not negatively impact the model's performance. In fact, it achieves a comparable performance to models that used domain-specific vocabularies and, in the case of the health domain, even led to a slight performance improvement. Therefore, the use of a continual vocabulary approach should not compromise the model's translation quality, and in addition, it improves its robustness against unseen new terms at no cost.

5.5 Discussion and Conclusions

In this chapter, we have explored the open-vocabulary problem in NMT, a fundamental challenge that arises due to the inability of models to effectively handle words or tokens not present in the training data. This issue becomes particularly pronounced as language evolves over time, introducing new words, domain-specific terminologies, and linguistic variations. Our study highlighted the trade-offs between the different vocabulary encodings, including the proposed QCB vocabularies, and emphasized their impact on translation quality, sequence length, and computational efficiency. Additionally, we studied how to effectively integrate new semantics into a pre-trained model, as well as practical solutions for using a continual vocabulary.

We began this chapter by defining the open-vocabulary problem and studying different vocabulary encoding strategies, ranging from byte- and character-level representations to subword- and word-level representations (Section 5.1). While low-level vocabularies offer higher granularity and robustness in handling rare or unseen tokens, they tend to produce longer sequences, which significantly increase the computational demands and hinder the learning of long-term dependencies. In contrast, high-level vocabularies introduce shorter and more manageable sequence lengths but often struggle with OOV problems. To bridge the gap between these approaches, we introduced Quasi-character-level encodings, a low-level strategy that extends character-level vocabularies with the most frequent character pairs. This approach dramatically reduces the sequence lengths associated with purely character-level encodings, mitigating the computational overhead and making them comparable to those of high-level

vocabularies while preserving the granularity needed to handle OOV words, offering a middle ground between the flexibility of low-level encodings and the efficiency of high-level ones.

In Sections 5.3 and 5.4, we turned our attention to incremental and continual vocabularies, investigating the requirements for seamlessly integrating new semantics into a pre-trained NMT model. Accordingly, a critical requirement for extending a pre-trained embedding set is ensuring that the new word embeddings align with the existing ones to maintain the semantic coherence within the same latent space. This involves matching dimensionality, using high-quality embeddings, and alignment mappings so that newly added tokens (embeddings) do not drift semantically from existing ones. Although incremental vocabularies can integrate new semantics into our models, they often rely on offline mechanisms to maintain a coherent semantic space between the old and new embeddings. In contrast, our approach to continual vocabularies addresses this issue by decoupling the vocabulary from the base model, and using a compositional embedding strategy. This approach exploits the subword information to construct new and semantically coherent embeddings for the unseen words that can be inferred from known subword units. By doing so, continual vocabularies enable an online vocabulary expansion without destabilizing the representation space, thus easing the handling of emergent terms.

Our experimental results emphasize the importance of aligning the vocabulary domain with the target domain for fixed-domain lifelong scenarios and, adopting a universal, non-domain-specific, and compositional vocabulary approach for open-domain lifelong scenarios. For instance, while domain-specific vocabularies often deliver great performance on in-domain data, they often struggle with generalization across domains. In contrast, the use of a continual vocabulary approach showed a greater domain adaptability, achieving more consistent and robust performance results across domains, while it was able to handle unseen words that can be derived from its known subword components.

In conclusion, effectively addressing the open-vocabulary problem in NMT requires the development of vocabulary strategies that balance the granularity and computational efficiency of the chosen encoding, enable the seamless integration of new semantics into a coherent latent space, and facilitate the inference of unseen words from the existing subword components.

Chapter 6

Mitigating the Catastrophic Forgetting in NMT

This chapter addresses the challenge of the Catastrophic Forgetting (CF) problem in Neural Machine Translation (NMT). Here, we explore different strategies to mitigate this phenomenon, including vocabulary effects, rehearsal methods, loss regularization, and parameter-efficient adaptation. These approaches seek to enable NMT models to learn continually from new data streams without sacrificing performance on previously learned tasks or, at the very least, mitigate these undesired effects.

6.1 Understanding the Catastrophic Forgetting

The Catastrophic Forgetting (CF) problem, also known as Catastrophic Interference, refers to the tendency of an artificial neural network to forget previously learned information upon learning new one (Mcclelland et al., 1995). This issue poses a major challenge in developing continual learning systems, where the goal is to enable Machine Learning (ML) models to adapt to new data while retaining previously acquired knowledge.

Addressing this problem is crucial in Artificial Intelligence (AI), but especially in applications like NMT, where language is continuously evolving with new words, expressions, and styles. Therefore, an NMT system must be able

to integrate all this new linguistic information while preserving its ability to translate previously learned content (e.g., words, domains, languages). However, due to the stability-plasticity dilemma (Hebb, 1949), achieving this balance is quite challenging, particularly given the complexity of the task and the high-dimensional parameter space of these models. Furthermore, this challenge is compounded by the fact that the Catastrophic Forgetting problem has not been extensively studied in Natural Language Processing (NLP) or, more specifically, in NMT, leaving a gap in effective solutions.

To tackle these challenges, we posed several critical questions that guided our research:

- How does vocabulary domain affect Catastrophic Forgetting problem? Can a continual vocabulary help to mitigate it?
- How effective is looking at minimal amounts of past data? Is there a way to achieve similar results without relying on more past data?
- In scenarios with no prior information, what approaches can we employ?
- Can we separate model-specific linguistic attributes (like language, style, and domain) from core semantic structures within the model?
- How much new knowledge can we incorporate into these separated components without compromising performance?

To answer these questions, we explored the CF problem from several areas: vocabulary domain, rehearsal methods, loss regularization, and parameter-efficient adaptation.

Consequently, by recognizing the vocabulary's influence on model retention capabilities, we first studied how the vocabulary domain and size impact the CF problem. Additionally, we assessed whether a continual vocabulary, constructed using compositional embeddings, could provide a more consistent embedding space that mitigates forgetting.

In addition to these vocabulary-based strategies, we investigated the effectiveness of a rehearsal strategy at varying levels of past information during sequential training, with the goal of minimizing the reliance on past data while maintaining a model performance.

Similarly, for scenarios where past data is limited or unavailable, we tried to combine a few-shot rehearsal strategy with oversampling and regularization methods to improve the model's retention under these constrained scenarios.

Finally, we investigated parameter-efficient adaptation techniques to decouple model-specific linguistic features from shared semantic structures that occur across many languages. Accordingly, we study the effectiveness of efficient task-switching strategies to adapt our NMT models to different languages, domains, and styles, using a minimal fraction of the model's parameters and without worrying about the CF problem. Also, we extend this approach to enable simple interactive adaptations. Furthermore, we proposed a gradient-based regularization technique for low-rank matrices to address the CF problem and to integrate new knowledge into these compressed matrices.

Together, these approaches offer a holistic strategy to mitigating the CF problem and enhance the continual learning capabilities in NMT models, while preserving their robustness in dynamic linguistic environments.

6.1.1 Definition and Impact on NMT

The Catastrophic Forgetting (CF) problem in neural networks can be formally defined as an optimization problem, where the objective is to minimize a loss function $\mathcal{L}$ over a sequence of tasks $\mathcal{T}_1, \mathcal{T}_2, \ldots, \mathcal{T}_K$ (Kirkpatrick et al., 2016; van de Ven et al., 2024). Thus, the goal is to find the parameters θ for the model f, where f_θ represents the neural network function (i.e. model architecture) parameterized by θ, that minimize the cumulative loss across all tasks:

$$\theta^* = \arg\min_{\theta} \sum_{n=1}^{K} \mathcal{L}(\mathcal{T}_n, f_\theta) \tag{6.1}$$

However, as training progresses sequentially over each task $\mathcal{T}_n$, the model parameters θ are updated without taking into account the loss of information from previous tasks, which results in a performance degradation on earlier tasks.

To address the catastrophic forgetting effects and be able to integrate new information, we hypothesize that a model, f_θ, could be viewed as a point within a high-dimensional space defined by its weight matrices and vectors, ultimately shaped by the data distribution on which it was trained. Thus, each task might define a region within this hyperdimensional space, with related tasks potentially overlapping (see Figure 6.1). Hence, the problem of mitigating catastrophic forgetting can be framed as adjusting the model's position within the overlapping region of these task-defined subspaces.

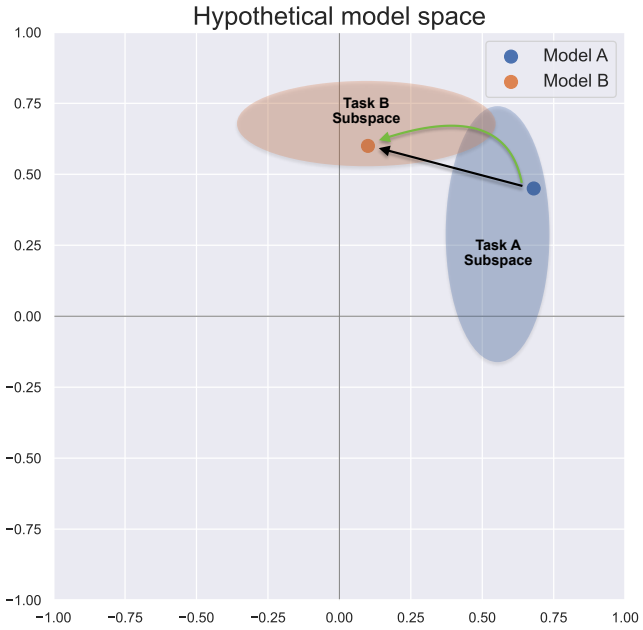


Figure 6.1: Hypothetical Model Space: Conceptualizing a model as a point within a high-dimensional space defined by its training data distribution, another model trained on a different but related distribution, could occupy a partially overlapping region in that space. To mitigate the CF, we aim to move the model within this overlapping region. The arrows indicate potential paths for the $f_{A,B}$ model; the green arrow passes through the overlapping area, and the black arrow bypasses it.

Formalizing the previous concept and, assuming an overlap between tasks exists, consider a model f_θ trained on task $\mathcal{T}_A$, resulting in a model f_A[1], we aim to adapt this model for a new task $\mathcal{T}_B$ while minimizing the performance degradation on the original task $\mathcal{T}_A$, producing a combined model $f_{A,B}$. In an NMT context, we view the learning of a new task as analogous to learning a new language, domain, or style.

In contrast, if no overlap between tasks exists, and we want to use a single set of parameters to eliminate the need for storing a separate model for each task, it may be feasible to either find the region in the space that minimizes the loss

[1] f_A, f_B, and $f_{A,B}$ denote models trained on tasks A, B, and both tasks respectively, for simplicity.

for both tasks or, at least, a transformation function that maps one model's point to another ($A \rightarrow B$). As in typical fine-tuning, we would typically start with a weight matrix W and after the fine-tuning process, we would end up with a weight matrix W'. For a non-overlapping new task or to avoid dealing with the CF problem, this transformation can be represented as an additional matrix ΔW, where the final matrix W' becomes:

$$W' = W + \Delta W \tag{6.2}$$

where ΔW represents the task-specific parameter updates. That is, ΔW_A and ΔW_B would represent the updates for tasks $\mathcal{T}_A$ and $\mathcal{T}_B$, respectively.

Therefore, our approach addresses two scenarios:

- **Tasks with an overlapping subspace**: We can attempt to adjust the model weights to move them into the shared region defined by these tasks. This approach could help the model retain performance on both new and previously learned tasks.
- **Tasks without an overlapping subspace**: We can find a transformation function to transform one set of weights into the other.

For overlapping tasks, we will focus on vocabulary domain, rehearsal, and loss regularization, discussed in Sections sections 6.2, 6.3.1 and 6.3.2, while for non-overlapping tasks, parameter-efficient adaptation strategies seem to provide a promising solution by circumventing the CF problem directly, as discussed in Section 6.4.

6.2 Vocabulary Influence on the Catastrophic Forgetting

In this section, we extend our previous experiments on NMT vocabularies discussed in Section 5.4, focusing specifically on how the vocabulary domain and size influence the CF problem. Here, our goal is to understand how different vocabulary strategies affect a model's ability to retain previously learned knowledge when adapting to new domains.

6.2.1 Impact of Vocabulary Domain and Size on Model Retention

To complement our previous findings on the impact of the vocabulary domain on model performance (see Section 5.4.1), we introduced a fine-tuning scenario to gain deeper insights into the effects of the catastrophic forgetting across various vocabulary domains, sizes, and test domains. Specifically, we investigated how catastrophic forgetting and cross-domain performance changes as a function of vocabulary size and domain (Carrión & Casacuberta, 2023b).

Accordingly, we used our previously trained models in the health domain (see Section 5.4.1), each one using different vocabulary domains and sizes, and trained on the SciELO-health dataset (Spanish-English). Then, we fine-tuned these models on the biological domain, denoted as $H \rightarrow B$, and evaluated their performance on three test domains: health, biological, and more general domain, the merged domain. Hence, our goal here was to visualize the effects of the catastrophic forgetting under these scenarios, as shown in Figure 6.2. It is worth pointing out that to avoid training biases, each training set was limited to 100k sentences[2].

The results in Figure 6.2 reveal that the vocabulary domain and size have a significant impact on the model's retention capabilities and cross-domain performance.

For example, in the first row, corresponding to trained models using a health-domain vocabulary, we observe that after fine-tuning the *health* model on the *biological* domain (columns labeled $H \rightarrow B$), the evaluations on the *health* test set deteriorate, while performance on the *biological* test set improves. This indicates that the model forgets previously learned information from the health domain when adapting to the biological domain, which is a clear sign of the CF problem (see red lines). Then, in the second row, which contains the models with the biological-domain vocabulary, a similar trend was observed. The model's performance on the health test set worsens after fine-tuning on the biological domain[3], which again emphasizes the impact of vocabulary domain on the retention of prior knowledge although not as strong as in the previous case as expected.

Interestingly, in the third row, which shows the models that were using the merged-domain vocabulary (combining both health and biological domains with the same vocabulary size), the catastrophic forgetting phenomenon was not present. That is, the performance on the health test set remained relatively

[2]The total number of training tokens was similar across datasets.

[3]This time results were a bit noisier due to the initial use of an out-of-domain vocabulary.

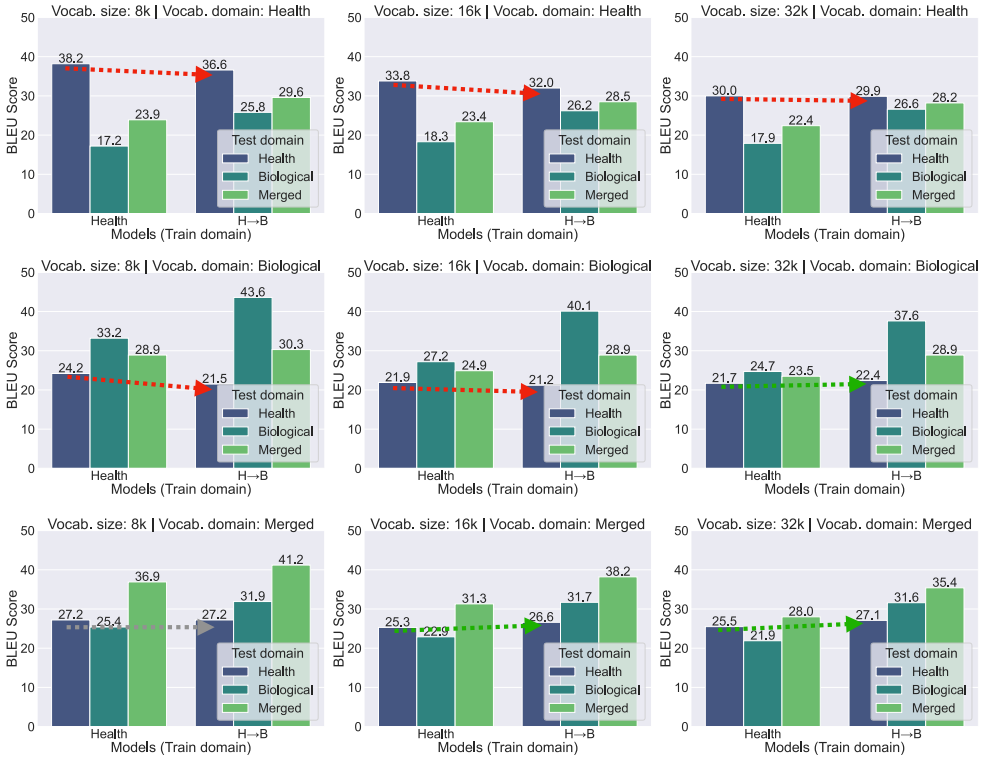


Figure 6.2: Comparison of a base (*Health*) and a fine-tuned ($H \rightarrow B$) model tested across three domains as a function of vocabulary size and domain alignment: Each model utilized a domain-specific vocabulary generated via BPE on one of the three training sets (domains), each one limited to 100k sentences to ensure balanced data volumes. The results suggest that by employing larger, general-purpose vocabularies (e.g., merged domains), we can partially mitigate the impact of the catastrophic forgetting, offering better cross-domain performance.

stable after fine-tuning on the biological domain, to the point of even improving its previous performance on the larger vocabularies (probably due to the additional training on a similar scenario). This finding suggests that the use of a more general vocabulary, more similar to the past domain, helped the fine-tuned model to retain the past domain knowledge more easily.

Additionally, we noticed that increasing the vocabulary size from 8k to 32k tokens appeared to slightly mitigate the effects of catastrophic forgetting in our setup, potentially because larger vocabularies captured a wider range of

lexical variations (even domain-specific terms), which may facilitate better generalization and knowledge retention across domains.

These findings imply that by using a more general vocabulary domain with larger sizes, we can mitigate the effects of catastrophic forgetting to some extent. Therefore, by using a broader spectrum of domain-specific terminology in the vocabulary, the model will be better equipped to retain past knowledge while trying to integrate new information from different domains. Interestingly, a vocabulary that may seem suboptimal for a given training set can still significantly enhance the model's ability to generalize and retain knowledge, thus helping to reduce the impact of catastrophic forgetting.

Similarly, to further validate the robustness of these observations, we repeated this experimentation using other datasets (e.g., Europarl, Multi30k), language pairs (e.g., en-es, en-pt, en-de, en-fr), and domains (e.g., CommonCrawl, NewsCommentary), which will be detailed in later sections as part of their own experimentation. Nonetheless, we anticipated that the results from these additional experiments were consistent with the initial observations presented in this section, thus, reinforcing the observation that the vocabulary domain and size play a significant role in mitigating catastrophic forgetting in NMT models. Consequently, these findings highlight the importance of performing a careful vocabulary selection in order to improve the model generalization across new tasks or domains.

6.2.2 Use of Continual Vocabularies: Compositional embeddings

As suggested in previous sections (see Section 5.4.1), increasing the overlap between domain vocabularies could improve the cross-domain consistency and help to mitigate the effects of the catastrophic forgetting phenomenon. To test this hypothesis, we adopted a continual vocabulary approach similar to the one described in Section 5.4.2. Specifically, our approach involved starting with an initial vocabulary V_0 and expanding it on demand with each new task or domain, leveraging the properties of universal compositional embeddings (Carrión & Casacuberta, 2023b). This approach can be formalized as:

$$V_K = V_{K-1} \cup V_{\text{new}} \tag{6.3}$$

where V_{new} represents the set of previously unseen words introduced by the new domain or task, and V_{K-1} is the vocabulary accumulated up to the previous task. The initial vocabulary V_0 can be assumed to be an empty set.

In practice, this meant training a model on the health domain using the continual vocabulary and subsequently fine-tuning it on a new biological domain. As new, previously unseen words appeared, we utilized the compositional properties of our vocabulary to dynamically generate new embedding representations on the fly. This experiment complements our previous research on the effects of the vocabulary on the CF problem (see Section 6.2.1) by adding a comparison between the effects of domain-specific vocabularies and our continual vocabulary approach (see Figure 6.3).

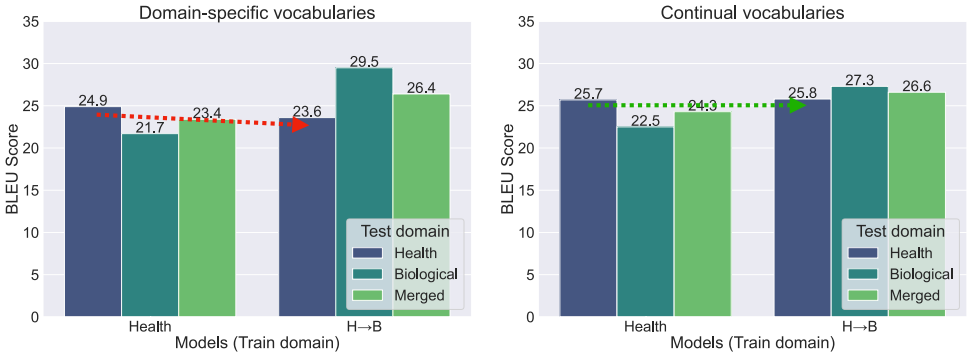


Figure 6.3: Performance comparison in BLEU points between models using domain-specific vocabularies (left) and those using continual vocabularies (right), before fine-tuning (Health) and after fine-tuning on a new domain ($H \rightarrow B$): The models with continual vocabularies shows a reduction in the standard deviation across test domains (Health, Biological, and Merged) and mitigated the effects catastrophic forgetting (Health vs. $H \rightarrow B$).

As shown in Figure 6.3, the performance of models using domain-specific vocabularies (baseline models) and those using continual vocabularies (our approach) were quite similar before fine-tuning (Health group). However, after fine-tuning these models on the Biological domain (see $H \rightarrow B$ group), we could observe significant performance differences across the test domains. That is, the standard deviation across test domains was reduced from 2.95 BLEU points with the domain-specific approach, to just 0.75 BLEU points when our continual vocabulary approach was used (see Table 6.1 for es-en).

Similarly, the effects of catastrophic forgetting were significantly smaller for the model using the continual vocabulary compared to the domain-specific vocabulary, to the point of even improving its previous performance measure. For instance, the health model using the domain-specific vocabulary experienced a loss of 1.3 BLEU points after fine-tuning (from 24.9 to 23.6), whereas the model

with the continual vocabulary not only maintained its previous performance but slightly improved it by 0.1 BLEU points (from 25.7 to 25.8). This indicates that more generic, universal continual vocabularies, can effectively mitigate the effects of catastrophic forgetting and improve the cross-domain consistency.

To further validate these findings, we repeated the experiment for another language pair (Portuguese-English), as shown in Table 6.1, noticing similar results and trends: (1) a reduction in the standard deviation of performance between domains, and (2), a mitigation of the CF problem when using continual vocabularies.

Table 6.1: Performance comparison in BLEU points of models fine-tuned with domain-specific versus continual vocabularies: The models fine-tuned with continual vocabularies showed a greater consistency across domains (indicated by the lower standard deviation) and mitigate the effects of the CF problem, as shown in the *Improvement after Fine-Tuning* column.

Vocabulary Type	Dataset	Train: H → B Test: H/B/M	Std. Dev.	Improvement after Fine-Tuning
Domain-Specific	SciELO (es-en)	23.6 / 29.5 / 26.4	2.95	−1.3 (24.9 → 23.6)
Continual	SciELO (es-en)	25.8 / 27.3 / 26.6	**0.75**	+**0.1** (25.7 → 25.8)
Domain-Specific	SciELO (pt-en)	24.5 / 30.8 / 27.4	3.15	−0.8 (25.3 → 24.5)
Continual	SciELO (pt-en)	26.3 / 29.1 / 27.8	**1.40**	+**0.2** (26.1 → 26.3)

These results indicate that adopting a universal continual vocabulary approach into our models leads to a more consistent cross-domain performance and, consequently, reduces the impact of catastrophic forgetting in NMT models. Also, despite the fact that the performance gains were relatively modest, this approach did not compromise the model's performance w.r.t a domain-specific vocabulary, as already discussed in Section 5.4.2. Furthermore, it enhances the model's ability to handle previously unseen terms without additional computational costs, making it a promising component of a lifelong learning strategy to improve the knowledge retention and cross-domain generalization in NMT models.

6.3 Strategies to Mitigate the Catastrophic Forgetting

One of the main goals for many lifelong learning systems is to enable ML models with the capacity to acquire new knowledge across different tasks without forgetting what they have already learned. In this section, we examine several strategies designed to mitigate the effects of the CF problem in NMT models. Specifically, we explore methods that either alleviate forgetting by

incorporating minimal rehearsal mechanisms or by performing an active few-shot regularization that re-weights past samples and penalizes weights that deviate too much from the original model.

6.3.1 Rehearsal Strategies

Rehearsal is a strategy to mitigate the catastrophic forgetting by interleaving previously seen data during the training of the new task. This approach ensures that the model consistently revisits past samples as it learns new information, thereby reinforcing existing knowledge and minimizing the loss of previously acquired capabilities.

In the following subsections, we investigate how varying amounts of past data, along with techniques such as re-weighting and regularization, influence the model's ability to retain its existing capabilities while effectively learning new linguistic patterns within a sequential NMT framework.

Sequential training with rehearsal

During our initial exploration of rehearsal strategies, we started by characterizing the catastrophic forgetting phenomenon as a function of the number of tasks learned and the ratio of past data incorporated during the learning of new tasks (Carrión & Casacuberta, 2022b).

Specifically, we conducted several experiments with Multilingual Neural Machine Translation (MNMT) models trained sequentially across multiple language pairs (tasks) while varying the amount of past data presented during training to measure its impact on the catastrophic forgetting. This focus on learning new language pairs, rather than learning new domains as before, was driven by the inherently greater complexity of multilingual tasks, which amplifies the effects of the catastrophic forgetting problem due to the smaller overlap between tasks (i.e., linguistic structures and vocabulary).

Accordingly, we started with a base model trained on English-Spanish (Task 1), we subsequently fine-tuned it on English-French (Task 2), followed by English-German (Task 3), and finally, English-Czech (Task 4) to saturate their weights. For each task, we trained multiple models with different proportions of interleaved past data during the learning of the new task. Finally, we also trained a MNMT model using all language pairs (*en-es/fr/de*) simultaneously, as a baseline comparison against the sequentially trained MNMT models.

The results, illustrated in Figure 6.4, show the effects of rehearsal during sequential training. Rows correspond to the task being learned, columns represent the model's performance on previous tasks while learning a new one, and the values (%) at the end of each line indicate the ratios of past data used per batch during the learning of the new task.

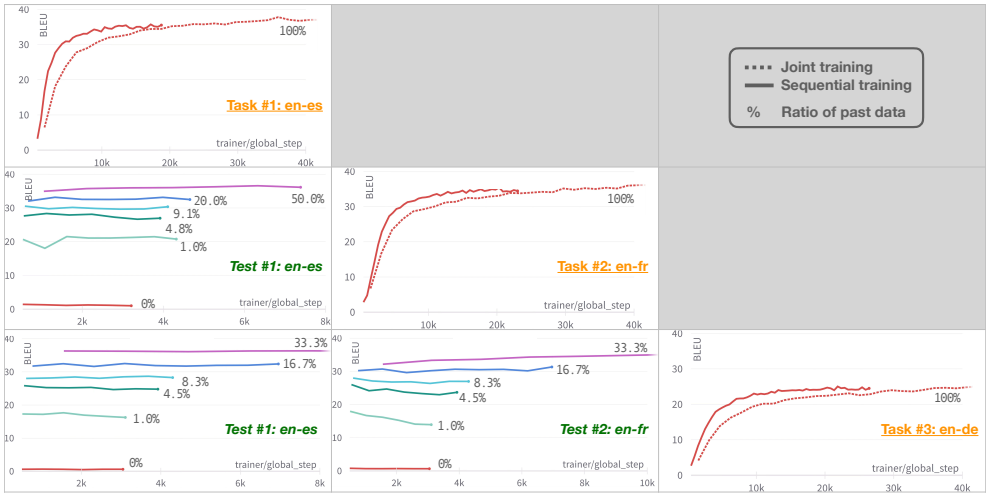


Figure 6.4: Impact of replaying different ratios of past data for mitigating catastrophic forgetting during a sequential MNMT training: Each row corresponds to the newly learned language pair (task), and each column shows how well the model retains performance (in BLEU points) on previously learned tasks. The percentages at the end of each line denote the ratio of past data per batch used to rehearse older tasks while learning a new one. As the figure illustrates, even with a minimal amount of data (e.g., 1%), the models can retain a significant amount of their performance on past tasks. Note: The red dashed lines represent the jointly trained baselines used as a comparison for the sequential runs, where these achieved a comparable performance in all cases.

These results show that after training the base models on the first task (i.e., English-Spanish), it achieved a BLEU score of 35. However, after fine-tuning the models on the second task (i.e., English-French) without including any past data, their performance on that previous task dropped to zero (as indicated by the red flat lines), a clear sign of catastrophic forgetting. In contrast, by incorporating a randomly sampled 1% of past data per batch, the models were able to retain up to 60% of their original performance on the first task after learning the second, under our experimental conditions. And then, when using 20% of past data, they preserved around 95% of their past performance. This pattern remained consistent as additional tasks (i.e., English-German and English-Czech) were learned. It is important to point out that, in our experimentation,

the past data samples were selected randomly from previously used training sets. While random selection does not explicitly optimize for coverage of linguistic phenomena, it ensures an unbiased revisit of past knowledge. However, more sophisticated strategies could, in principle, select better data that represent a diverse set of previously learned linguistic features, potentially improving the mitigation of catastrophic forgetting problem.

Interestingly, during the learning of new tasks, the models' performance on previous tasks remained relatively stable overall. This was unexpected and especially noteworthy because we anticipated a continuous decline in all tasks due to the catastrophic forgetting and network saturation problems. However, when comparing the sequentially trained models with the jointly trained baselines (all language pairs), no significant differences in performance were found (as depicted by the red dashed lines in Figure 6.4). In contrast, it was not until we added a fourth task (English-Czech; See Figure 6.6) that more pronounced signs of the catastrophic forgetting started to appear, supporting the hypothesis that as the model approaches its capacity limit, it tends to forget information more rapidly despite the addition of past data.

Furthermore, it is important to consider that, in addition to the network saturation problem, the selected language pairs shared certain linguistic characteristics (Indo-European languages), which likely make them partially transferable and more resilient to the catastrophic forgetting problem. In contrast, for completely unrelated languages (e.g., Spanish-Chinese), this network saturation problem is expected to occur more quickly, as closely related languages or those with shared linguistic roots allow previously learned representations to be reused more effectively, providing a temporary buffer against severe forgetting.

Nonetheless, it seems that as long as the models have a sufficient capacity along with a mechanism to manage both new and old information (e.g., rehearsal, regularization, complementary learning, ...), their performance does not appear to differ significantly regardless of whether they are trained sequentially or jointly.

These results suggest that maintaining satisfactory performance on previous tasks does not require using all the previous data, or at least not a large amount of them. Instead, even by adding tiny amounts of past data during the learning of the new task, we can significantly reduce the effects of catastrophic forgetting. Hence, this approach seems to be particularly useful for extending the number of domains or language pairs supported by NMT model in a lifelong learning scenario, especially when access to past data is limited or difficult to obtain.

Learning to rehearse with minimal data

In this section, we decided to explore the limits of our previous findings by trying to maximize the performance across all tasks while minimizing the amount of past data needed. To do so, we conducted a similar experiment to the one described in the previous section 6.3.1, but this time, oversampling past information through a task-weighting coefficient (Carrión & Casacuberta, 2022b).

Specifically, there are two common ways to approach this problem: (1) Oversampling on the training data, and (2), oversampling on the delta (Error Signals). For the first case, if the new task had 10,000 samples and the previous tasks had 1,000 and 2,000 samples, we would assign a weighting coefficient of 1.0 for the first task, and 10.0 and 5.0 for the second and third tasks, respectively[4]. For the second case, we penalize the sentences in the batch based on a given coefficient vector w_t, which ultimately depended on the specific task associated with that sentence. Moreover, we end up experimenting with a customized version of a focal loss that applies a scaling factor to the error signal for hard samples to give them more weight during training. This can be seen in the following equation:

$$\mathcal{L} = \sum_{n=1}^{K} \frac{w_n}{M} \sum_{i=1}^{M} (1 - p_i)^{\gamma} \mathcal{L}_{\text{CE}}(y^{(i,n)}, \hat{y}^{(i,n)}) \tag{6.4}$$

where:

- $\mathcal{L}$ is the cumulative loss aggregated over all tasks and training instances.
- $\mathcal{L}_{\text{CE}}$ is the Cross-Entropy (CE) loss.
- K is the total number of tasks.
- M is the total number of source-target pairs.
- $y^{(i,n)}$ and $\hat{y}^{(i,n)}$ represent the true and predicted sentences, respectively, for the i-th training instance in the n-th task.
- w_n is the task-specific weighting factor for the n-th task, manually tuned or empirically learned, to balance the contribution of each task to the

[4]These weight values are simply illustrative and were chosen to enforce the idea that each task was balanced w.r.t. the number of samples and/or their contribution.

overall loss. A common heuristic involves setting w_n inversely proportional to the size of the task data.

- p_i is the average predicted probability for the correct tokens in the i-th sentence, providing a measure of the model's confidence in its predictions. The values of p_i are derived directly from the softmax outputs.
- γ is a focusing parameter that determines the extent to which harder examples (low p_i) are emphasized during training. Its value is typically set via hyperparameter search, with $\gamma = 2$ being a widely used default.

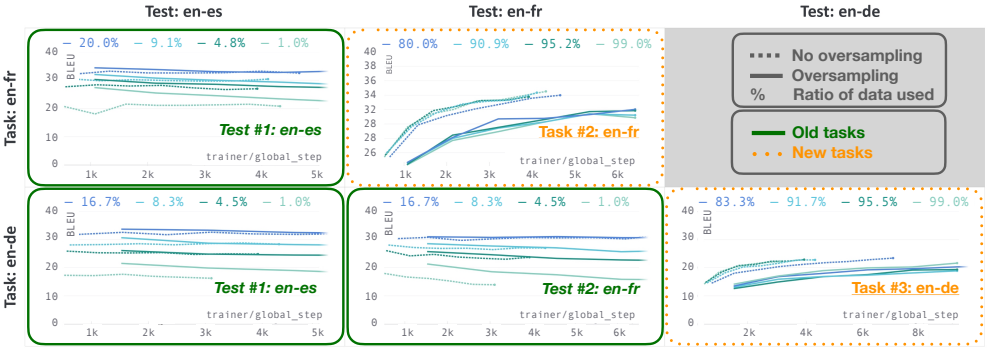


Figure 6.5: Performance comparison in BLEU points of oversampling and non-oversampling strategies for mitigating catastrophic forgetting in sequential training: The oversampling strategy on minimal past data (1%) improved past task retention by up to +4 BLEU points, while higher past data ratios (4.5%+) showed no significant difference w.r.t non-oversampled methods.

In Figure 6.5, we can see the results from this experiment, comparing the oversample and non-oversample methods on a sequential training setup (in green the old task, and in orange the new tasks). As a result, the major benefit from using an oversampling strategy was achieved when minimal amounts of past data were used (1%), obtaining up to +4pts of BLEU more than the non-oversampled approach, which seems to indicate that this oversampling strategy helps to retain past knowledge slightly better than the non-oversample methods. However, this was at the expense of a slightly worse learning on the new task than their counterparts. In contrast, when higher amounts of past data were used (4.5% or more), no significant differences were found between them.

We also experimented with multiple task-weighting coefficients, and from our experimentation results, we noticed that higher task-weighting values usually

made our NMT models more prone to overfit on past tasks. To counteract this, we tried learning these task-weighting coefficients dynamically, and then we applied an additional term (focal term) to the loss, in both cases, achieving similar results as the ones presented here. Consequently, we reached a sort of a *pareto frontier* where the learning improvements in one task were at the expense of others.

6.3.2 Regularization Strategies

In contrast to rehearsal-based methods that rely on explicitly revisiting past samples, regularization-based strategies aim to mitigate the catastrophic forgetting problem by penalizing changes to the model's parameters so that they remain close to their previous values.

In this section, we explore how regularization, combined with a minimal rehearsal strategy, helps to retain most of the past knowledge while learning a set of sequential tasks.

Few-shot rehearsal: Combining rehearsal with oversampling and regularization

As discussed in the previous Section 6.3.1, if we increase the number of tasks to be learned and do not increase the capacity of the model, sooner or later, the model will reach a saturation point, where regardless of the amount of new and old training data, we will have to decide which information is forgotten in favor of the new one.

Consequently, in this section, we try to selectively forget the less relevant knowledge for the past tasks in favor of incorporating as much relevant knowledge as possible for the new tasks (Carrión & Casacuberta, 2022b). Specifically, we decided to complement the strategies defined earlier (i.e., minimal rehearsal and delta oversampling) with a customized loss function that tries to minimize the forgetting of previously learned tasks by actively re-weighting past samples and penalizing weights that deviate too much from the original model (past weights).

That is, initially, all tasks should contribute equally to the loss regardless of the amount of past data available, but then, these weights would be slightly modified to ease the convergence of the new task (re-weighting). This re-weighting was calculated using two strategies: (1) Averaging the probability of the j-th token in the i-th sentence being correct for a given task, and (2), adding

a task-parameter to maximize the overall learning in all tasks. In addition to this, we defined the loss so that errors could be penalized more severely on past tasks than on new ones through an exponential hyperparameter so that we could have more control over the forgetting effects. Finally, we wanted to penalize changes in weights that are assumed to be relevant for the past tasks but are not for the new task. To do so, we added a regularization term, based on the knowledge distillation loss derived by (Hinton et al., 2015), to control for these deviations in past tasks with respect to the previous version of the same model, which is presumed to be better in past tasks due to the effects of the catastrophic forgetting phenomenon.

This loss function is defined in the following equation:

$$\mathcal{L} = \alpha \cdot \mathcal{L}'(y_{\text{new}}, \hat{y}_{\text{new}}) + \beta \cdot \mathcal{L}'(y_{\text{past}}, \hat{y}_{\text{past}}) + \gamma \cdot \mathcal{L}_{\text{KD}}(z_{\text{s}}(x_{\text{past}}), z_{\text{t}}(x_{\text{past}})) \quad (6.5)$$

where:

- $\mathcal{L}'$ is the focal cross-entropy loss with oversampling derived from the one described in Section 6.3.1
- $\mathcal{L}_{\text{KD}}$ is the knowledge distillation loss of (Hinton et al., 2015). It is computed by comparing the current model's output distribution on the rehearsal data x_{past} to the predictions made by a previously saved version of the model (teacher model). This term helps to retain past knowledge by ensuring that the current model does not deviate significantly from the teacher's predictions on earlier tasks.
- $y_{\text{new}}, \hat{y}_{\text{new}}$ represent the true and predicted target pairs from the current (new) task data. The model is trained to minimize the error in these samples.
- $y_{\text{past}}, \hat{y}_{\text{past}}$ represent the true and predicted target pairs sampled from the previously learned tasks.
- $z_{\text{s}}(x_{\text{past}})$ results in the set of logits (unnormalized scores) produced by the current model (student) after processing x_{past}.
- $z_{\text{t}}(x_{\text{past}})$ results in the set of logits (unnormalized scores) produced by the previous model (teacher) after processing x_{past}.

- α, β, and γ are hyperparameters that control the contribution of the loss terms for the current task, rehearsal samples, and distillation loss, respectively.

Accordingly, we extended the experiment presented in the previous section 6.3.1, where we trained a base NMT model (i.e., English-Spanish) to learn new language pairs sequentially with few-shot rehearsal (i.e., English-French, English-German), but this time, we added another language pair (i.e., English-Czech) to test this strategy and hope to reach a saturation point. In addition to this, we incorporated the training runs that resulted from combining the previously defined strategies (few-shot and oversampling) with the custom focal loss function that we have defined in this section. This can be seen in Figure 6.6, where the solid red lines represent the previous strategy, and the blue lines represent the multiple training runs (not cherry-picked) from running the current strategy.

It is worth noting that the results reported in BLEU points in Figure 6.6 correspond to evaluations on a validation set performed during training. By the end of the training, the validation results were practically the same as those achieved on the test set, given the fact that both sets were randomly sampled and excluded from the training dataset. However, by showing the validation scores during training, we highlight the learning dynamics and the rate of adaptation rather than relying on a single test score, which could be more susceptible to noise. Again, the few-shot rehearsal only added a minimal amount of past data (1%) to alleviate the forgetting (same ratio as in the previous experiment -red lines-).

As illustrated in Figure 6.6 (tasks en-es, en-fr, and en-de), the proposed model (blue lines) demonstrated a significant reduction in forgetting compared to the previous and naïve approach (red lines). Interestingly, on the first two tasks (en-es and en-fr), our strategy not only mitigated the forgetting but also improved base performance. This improvement was likely due to the shared linguistic structures between the new language pair (en-fr) and the previous one (en-es), effectively leveraging the additional implicit training data. Furthermore, the model still had sufficient capacity to ignore potential network saturation problems. For the third task (en-de), while some performance degradation occurred, probably due to the first signs of network saturation problems, these were small, and the loss was considerably smaller than the one observed with the reference model (red line). Finally, with the addition of a fourth task (en-cs), the improvements in retaining past knowledge (tasks en-es, en-fr, and en-de) came at the expense of a slightly slower learning rate for the new task

(en-cs) and a slightly worse performance relative to the naïve approach. This trade-off was likely due to the additional control mechanisms and the amount of previously encoded knowledge (four tasks) within the network.

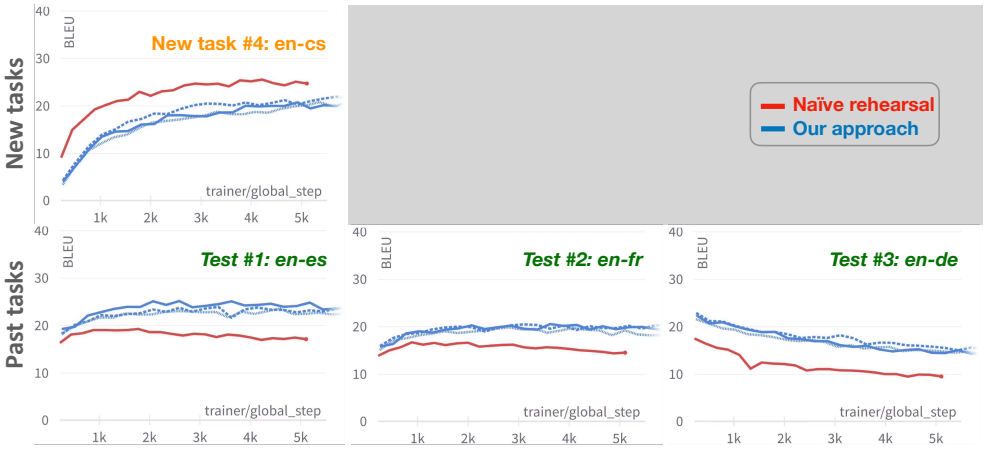


Figure 6.6: Effects of the catastrophic forgetting in BLEU points after sequentially learning four language tasks (en-es/fr/de/cs): Our approach (focal loss with a few-shot rehearsal; blue lines) demonstrated a superior knowledge retention on past tasks (es/fr/de) compared to a naïve rehearsal strategy (red line). However, this improved retention came at the cost of a slightly slower convergence rate on the new task (en-cs) due to the additional control mechanisms.

Therefore, this new loss, in addition to a minimal rehearsal strategy combined with an efficient oversampling mechanism, presents itself as a highly simple and practical approach to not only mitigate the effects of the catastrophic forgetting phenomenon, but as a technique that is able to incorporate new knowledge into a fairly saturated neural network, with almost no additional computational cost[5].

6.4 Parameter-Efficient Adaptation

As discussed in Section 6.1.1, the Catastrophic Forgetting problem can be formally defined as an optimization problem over a sequence of tasks $\mathcal{T}_1, \mathcal{T}_2, \ldots, \mathcal{T}_K$. However, rather than trying to minimize a loss function $\mathcal{L}$ over a set of potentially overlapping tasks, we can attempt to decouple the core semantic structures of our models (i.e., the model's general, language-agnostic semantic capabilities)

[5]Some previous task-dependent values need to be precomputed beforehand.

from the task-specific linguistic components such as domains and styles (maybe even language). In other words, we argue that the model's underlying ability to translate from one language to another could be partially separated from the parameters that capture particular domain-specific terminology or stylistic nuances. By doing so, each domain or style specialization can be considered as a distinct (non-overlapping) task with its own dedicated parameters. This separation allows us to adapt to new tasks without extensively altering the core translation components, effectively mitigating the catastrophic forgetting.

The most straightforward way to achieve a task-specific decoupling solution would be to train a new specific layer on top of the base model. However, while this approach might work for simple cases like similar domains or styles, it lacks the representational power to change more complex aspects, such as the language of a given model. Alternatively, we could fine-tune a new model for each individual task starting from a generic base model. However, this method would be highly inefficient due to the significant amount of computational resources required, both in training time, memory usage, and storage capacity. To address this problem, we focused our efforts on Parameter-Efficient Fine-Tuning (PEFT) solutions, which allow us to use a minimal number of additional parameters to keep a shared base model across many tasks while avoiding the redundancy and inefficiency of maintaining different full models for each task (Houlsby et al., 2019).

Consequently, by leveraging PEFT methods, we aim to integrate task-specific adaptations without compromising the overall scalability and performance of NMT models (Carrión & Casacuberta, 2024). Specifically, we seek to reduce the computational costs and parameter count needed for task-specific fine-tuning while decoupling most of the task-specific parameters from the base model. Among these methods, Low-Rank Adaptation (LoRA), by Hu et al., 2021, stands out as a particularly effective approach because, instead of inefficiently storing all the weights of the network after fine-tuning, it freezes the model's original parameters and learns the low-rank matrices needed to approximate the changes required in the high-dimensional weight matrices in a much more efficient way. That is, in a traditional fine-tuning approach, we would typically start with a weight matrix W, and after the fine-tuning process, we would end up with an updated weight matrix W'. This means that, as explained in Section 6.1.1, the changes in the matrix W can be represented by a second matrix ΔW such that the final matrix W' can be computed as:

$$W' = W + \Delta W \tag{6.6}$$

With this in mind, LoRA proposes a matrix factorization approach to approximate this weight update matrix $\Delta W \in \mathbb{R}^{p \times q}$ by two low-rank matrices X and Y, such that $\Delta W \approx XY^\top$, while leaving the original W fixed. As a result, the LoRA-based update is formulated as follows:

$$W' = W + XY^\top \tag{6.7}$$

where $X \in \mathbb{R}^{p \times r}$ and $Y \in \mathbb{R}^{q \times r}$ are the low-rank matrices, with $r \ll \min(p, q)$ being the rank of the adaptation.

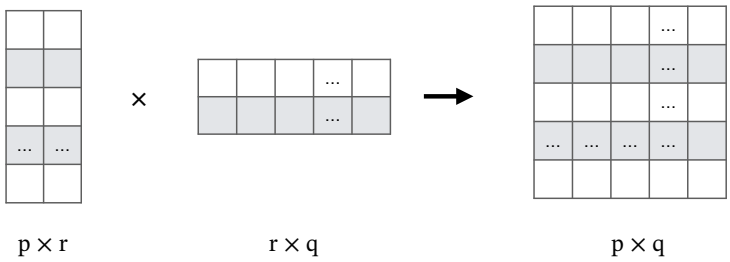


Figure 6.7: Visualization of a Low-Rank Matrix Decomposition: A large parameter matrix $W \in \mathbb{R}^{p \times q}$ is adapted using two smaller matrices $X \in \mathbb{R}^{p \times r}$ and $Y \in \mathbb{R}^{q \times r}$. The decomposition significantly reduces the number of parameters required for task-specific fine-tuning from pq to $r(p + q)$, with $r \ll \min(p, q)$.

This is illustrated in Figure 6.7, where the low-rank decomposition reduces the parameter space from pq to $r(p + q)$, thereby significantly decreasing the computational and storage overhead. Hence, by maintaining a shared base model, simplified as a weight matrix W, and selectively applying these low-rank updates $XY^\top$, we facilitate the rapid and efficient adaptation of NMT models. This approach allows for a task-switching strategy, thus enabling the model to seamlessly adapt to different languages, domains, or styles while preserving its core structure. Consequently, for each task $\mathcal{T}_n$ (where $1 \leq n \leq K$), we update the model weights as follows:

$$W'_{\mathcal{T}_n} = W + (XY^\top)_{\mathcal{T}_n} \tag{6.8}$$

Building upon these foundations, in the following subsections we will explore how to leverage these techniques to:

1. **Enable efficient task-switching strategies**: We will examine the effectiveness of incorporating a low-rank strategy into the NMT architecture to

efficiently switch the language or domain expertise of our models without compromising performance on previous tasks and using only a fraction of the model's parameters.

2. **Perform interactive domain adaptations**: By employing a calibrated Mixture of LoRA Experts (MoLE), we will adapt our NMT models interactively to different domains and styles in a plug-and-play manner, eliminating the need for additional fine-tuning (Jacobs et al., 1991).

3. **Incorporate new knowledge into these low-rank matrices**: We will discuss how to integrate new information into these low-rank matrices using a gradient-based regularization strategy specifically designed to address the CF problem in this scenario.

6.4.1 Efficient Task Switching

In this subsection, we explore the effectiveness of the low-rank adaptation method for fine-tuning NMT models in a parameter-efficient manner, focusing on efficient task switching between different domains, styles, and languages. Specifically, we study two main scenarios:

1. **Domain-specific model adaptation**: We aim to adapt a generic NMT model to boost its performance on specific domains using a minimal fraction of its parameters.

2. **Efficient language-switching**: We aim to change the language-pair that a pre-trained NMT model can translate to, to a new language-pair, or at least, to improve its performance on known language-pair efficiently.

Domain-specific model adaptation

For our first experiment, we trained a generic English-Spanish NMT model on a balanced mixture of sentences from various domains (e.g., Health, Biological, Legal, and Europarl) (Carrión & Casacuberta, 2024). The details of these datasets are described in Section 4.1. This controlled mixed-domain dataset allowed us to ensure that the model was exposed to a diverse yet balanced set of domains, reducing potential biases from overexposure to any single domain. Additionally, the balanced nature of the dataset enabled us to maintain a compact and well-calibrated model, making it ideal for our experimentation.

Next, we fine-tuned this base model using the LoRA technique with various ranks ranging from 1 to 256, corresponding to parameter counts from 17,472 to

2,236,416. This range allowed us to study the performance improvements in relation to the number of parameters used during the fine-tuning. Additionally, we fine-tuned a full-parameter model for each domain to serve as a comparison baseline for the low-rank models. In addition to this, we also included the performance of the base model (without fine-tuning) on each domain as a reference point.

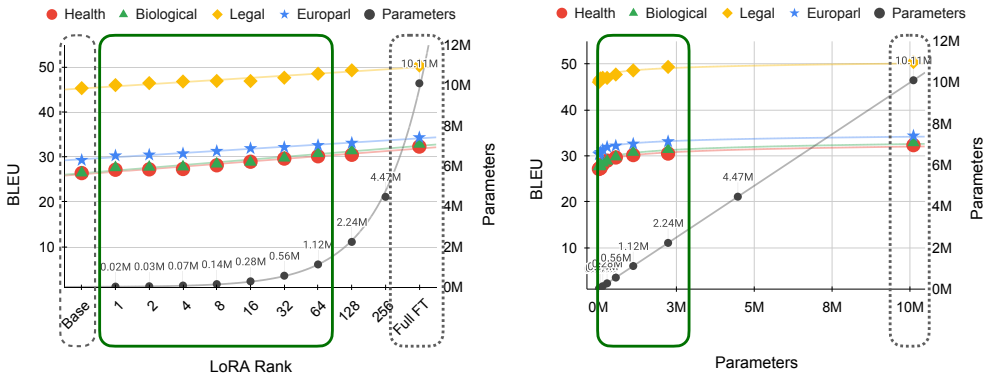


Figure 6.8: Comparative analysis of model performance vs. parameter count: This figure shows the BLEU score improvements achieved through a low-rank fine-tuning strategy across various domains. The results indicate that the LoRA fine-tuning approach follows a logarithmic trend, enabling substantial performance gains with only a minimal increase in parameters, which allows the model to reach performance levels comparable to traditional full-parameter fine-tuning methods while using a fraction of its parameters. Note: M = millions.

In Figure 6.8, we can see the performance improvements achieved through the low-rank fine-tuning strategy across different domains. The left figure shows the linear relationship between BLEU score improvements and the number of parameters, while the right plot presents the same data on a logarithmic scale. As observed, increasing the number of parameters through higher ranks (i.e., more parameters) led to logarithmic performance improvements across all domains. In contrast, smaller ranks (i.e., fewer parameters) yield to substantial performance improvements, exponential, relative to the parameter increase. Hence, the use of larger ranks seems to provide diminishing returns compared to smaller ranks.

Accordingly, in Table 6.2, we have a detailed breakdown of the performance improvements per domain relative to the base model as a function of the rank (i.e., number of parameters) used during fine-tuning. From these results, we observe that even by using as little as 0.17% of the model's parameters (rank 1),

we achieved an average of 16.20% relative improvement w.r.t the full-parameter Fine-Tuning (FT) performance (3.04% absolute improvement). Then, with just 1-5% of its parameters, we reach 30% to 50% of the full fine-tuning performance (6.13% to 9.93% absolute improvement). And with a rank of 64, which uses just 11.06% of its parameters, we achieved a 65.19% of relative performance improvement (12.23% absolute improvement).

Table 6.2: Relative performance improvements per domain with low-rank fine-tuning: Smaller ranks achieve performance improvements close to full fine-tuning with a fraction of its parameters. Note: K = thousands, M = millions.

LoRA Rank	Params used	Health	Biological	Legal	Europarl	Average
1	17.4K (0.17%)	2.82%	4.44%	1.50%	3.39%	3.04% ± 1.23%
2	34.9K (0.35%)	3.17%	4.87%	2.49%	4.04%	3.64% ± 1.04%
4	69.8K (0.69%)	3.48%	5.08%	3.27%	4.99%	4.20% ± 0.96%
8	139.7K (1.38%)	6.59%	7.53%	3.54%	6.85%	6.13% ± 1.77%
16	279.5K (2.77%)	9.60%	9.09%	3.50%	8.93%	7.78% ± 2.87%
32	559.1K (5.53%)	12.25%	12.55%	5.13%	9.80%	9.93% ± 3.43%
64	1.1M (11.06%)	14.22%	16.33%	7.09%	11.27%	12.23% ± 4.00%
128	2.2M (22.13%)	15.47%	18.69%	8.78%	13.04%	14.00% ± 4.18%
FT	10.1M (100.00%)	22.40%	24.24%	11.15%	17.25%	18.76% ± 5.88%

Style-specific model adaptation

To further investigate the effectiveness of low-rank strategies in adapting NMT models to specific linguistic variations, we conducted an experiment focused on style adaptation, seeking to adapt an NMT model to different levels of formality (Carrión & Casacuberta, 2024). For this purpose, we created a synthetic dataset using *DeepL* ("DeepL Translator", 2023), maintaining identical source sentences to reduce the encoding variability while presenting the target sentences in three different formality styles: neutral (Multi30K), formal (Multi30K-Formal), and informal (Multi30K-Informal).

To do so, we began by training an NMT model from scratch on the Multi30K dataset, which can be considered as a neutral level of formality. Next, we applied a low-rank strategy to fine-tune the decoder on the Multi30K-Formal and Multi30K-Informal datasets separately, using various ranks to evaluate how effectively low-rank matrices could capture style variations with minimal parameters. For comparison, we included a baseline of a full-parameter fine-tuned model. Our decision to fine-tune only the decoder was intended to isolate the model's ability to adapt new translation styles (outputs) without affecting its encoding capabilities (inputs), thereby preserving the original encoder to

have a more controlled evaluation of the style adaptation. Hence, by sharing identical source sentences across datasets and using the same encoder in all the evaluations, this setup provided an isolated assessment of the low-rank fine-tuning effectiveness to adapt the translation style.

The results shown in Table 6.3 present the BLEU scores across the Neutral, Informal, and Formal domains. Interestingly, even with just 8,736 parameters (rank 1), roughly 17KB in 16-bit precision, we were able to effectively bias the translation style of our NMT models (with around 10M parameters). Although these improvements may appear modest, it is important to consider that the original model already achieved a high translation quality of around 50pts of BLEU. Therefore, and given the fact that stylistic adaptation aims to subtly modify the output without altering the original meaning, we expect performance gains within this range.

Table 6.3: Performance on the Multi30K-Formality dataset: BLEU scores of NMT models using low-rank ranks adaptation for the Neutral, Informal, and Formal domains.

		Neutral→Informal			Neutral→Formal		
Rank	**Parameters**	**Neutral**	**Informal**	**Formal**	**Neutral**	**Informal**	**Formal**
1	8,736	49.98	**50.66**	50.05	50.29	50.32	**50.31**
2	17,472	50.15	**50.77**	50.44	50.38	50.38	**50.43**
4	34,944	49.97	**50.70**	50.17	50.39	50.53	**50.76**
8	69,888	50.16	**50.97**	50.44	50.67	50.84	**51.09**
16	139,776	50.14	**51.01**	50.37	50.80	50.95	**51.31**
32	279,552	50.42	**51.27**	50.71	50.84	51.14	**51.33**
64	559,104	50.66	**51.36**	50.80	51.35	51.59	**51.73**
FT	10,176,576	51.24	**51.60**	51.40	50.72	51.47	**51.94**

Efficient language-switching

Next, we investigated whether a low-rank adaptation approach could have enough representational power to enable an effective language-switching strategy in NMT (Carrión & Casacuberta, 2024). To test this, we designed two experiments:

1. **Boosting performance on known language pairs (barely seen)**: We first trained a base multilingual NMT model on a minimal multilingual dataset of four language pairs (English-Spanish, -French, -German, -Czech), with only 25k sentences per language. This training allowed the model to develop a basic understanding of each language, separating their

linguistic features before applying the aforementioned language-specific adapters. Then, we trained a set of low-rank adapters for each language pair individually, in order to boost the model's performance in each specific language.

2. **Incorporating new, unseen language pairs**: We tried to incorporate two completely new language pairs (English-Italian and -Portuguese) into a pre-trained multilingual NMT model using low-rank adapters. This experiment examined whether a low-rank adaptation strategy was powerful enough to effectively incorporate new unseen language pairs from scratch, using a minimal amount of parameters.

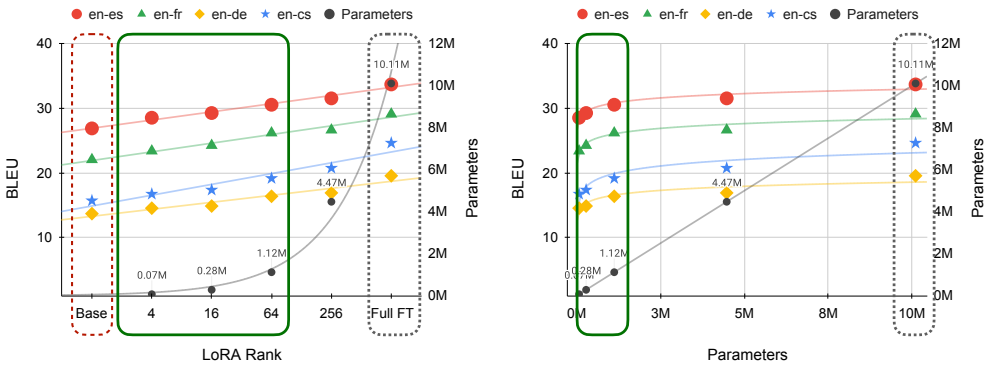


Figure 6.9: Efficient performance boosting for multilingual NMT models: A low-rank adaptation can significantly improve the performance of a minimally trained multilingual NMT model, using a minimal number of parameters. This trend follows a logarithmic curve, with the initial additional parameters yielding exponential gains, followed by diminishing returns on the higher end. Note: M = millions.

In Figure 6.9, we show the results of the first experiment, which explores the language boosting case. Similarly to the domain adaptation experiment, the low-rank adapters significantly boosted the performance in the known languages, following a logarithmic trend until they matched the performance on the fine-tuned full-parameter model. This suggests that even with just a few extra parameters, we can achieve substantial performance gains, albeit with diminishing returns at the higher parameter levels.

Then, for the second experiment (incorporating unseen languages), we aimed to add two completely new and unseen language pairs into a pre-trained multilingual NMT model. Since the language pairs were new, we used a pre-trained MNMT model as a foundational model to ease the discrimination between the different languages. As a result, the curves in Figure 6.10 show

that low-rank adapters can successfully incorporate these new language pairs into a pre-trained NMT model. However, incorporating new languages was a significantly more complex task than a simple domain adaptation or language boosting, as it required of higher ranks (i.e., more parameters) to achieve comparable results. Again, the parameter-performance function followed a logarithmic trend, consistent with previous experiments. For example, with 11.06% of the model's parameters (rank 64), we achieved up to 72.94% of full-parameter fine-tuning performance (e.g., English-Portuguese).

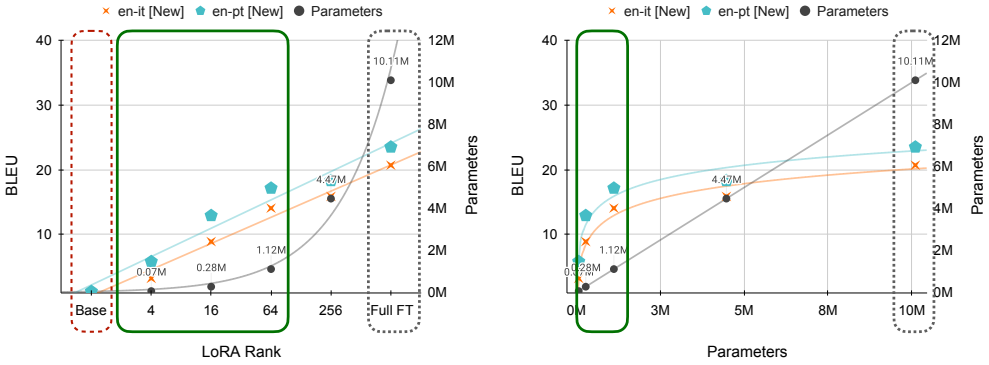


Figure 6.10: Incorporating new unseen language pairs into a MNMT model: A low-rank adaptation strategy can extend the supported language pairs of a MNMT model. However, this requires more parameters than a simple domain adaptation or language boosting due to the complexity of the task. Yet, the parameter-performance curve follows a logarithmic trend (exponential gains first, diminishing returns later). Note: M = millions.

Limitations of LoRA for task-switching

One of the main limitations of the low-rank approach is that it relies on a form of matrix factorization to approximate the model weight updates, which inherently introduces a lossy compression of knowledge. The extent of this compression loss will depend on the rank of the matrices used. That is, higher ranks will reduce the compression loss but also require more parameters (i.e., less efficient). Therefore, by learning efficient low-rank components (i.e., fewer parameters), these adapters may struggle to capture complex non-linear dependencies compared to a full-parameter fine-tuning approach.

This limitation can result in a suboptimal performance for complex tasks that demand intricate feature interactions or involve highly variable data distributions. Furthermore, reconstructing the target weight matrices from

low-rank components adds another layer of computational complexity, which can extend training times.

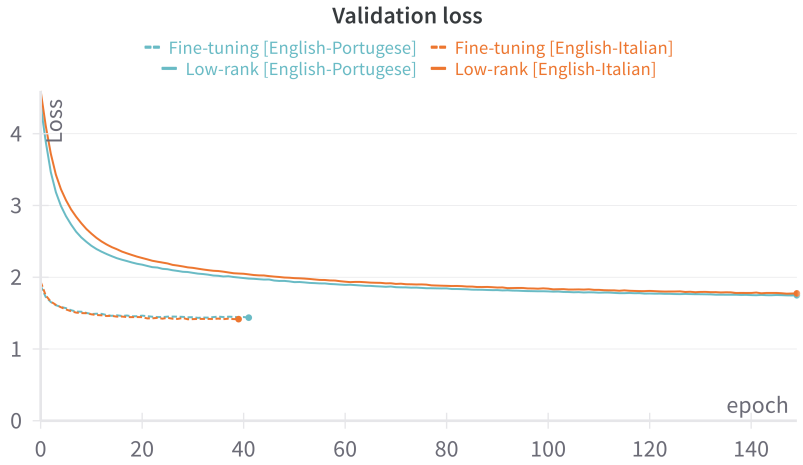


Figure 6.11: Comparative analysis of training times between low-rank and full-parameter fine-tuning approaches: The solid lines represent the low-rank approach, which exhibits longer training times compared to the full-parameter approach (dashed lines) due to the complexity of capturing non-linear dependencies through a matrix factorization process in smaller parameter spaces. In this case, the low-rank approach used 44.26% (rank 256) of the full-parameter model's parameters.

As shown in Figure 6.11, the low-rank fine-tuning converges more slowly than the full-parameter fine-tuning, thus requiring more computation (i.e., training time) to reach a comparable performance level. This issue is especially noticeable at lower ranks, where the reduced parameter space severely limits the model's ability to encode necessary knowledge effectively.

Discussion and Implications

The experiments conducted in this section demonstrate that a low-rank adaptation strategy such as LoRA provides an effective and parameter-efficient method to perform task-switching in NMT models, whether adapting to new domains, styles, or even languages.

By using a minimal fraction of the model's parameters required for fine-tuning a full-parameter model, we can achieve substantial performance improvements through these adapters, making low-rank adaptation strategies a practical

solution to address the Continual Learning problem in NMT. Furthermore, techniques that combine parameter-efficient methods with quantization, such as QLoRA (Dettmers et al., 2023), have recently emerged, offering the potential to further reduce the resources needed, although not explored here, they present a promising direction for future work.

However, it is important to consider the trade-offs in terms of potential limitations in capturing complex dependencies, weight saturation problems, lossy compression of knowledge, and increased training times due to the complexity of the task.

6.4.2 Interactive Domain Adaptation

Building upon the low-rank strategies discussed earlier (see Section 6.4), in this section, we explore the potential of interactively adapting our pre-trained NMT models to different domains and styles using parameter-efficient methods (Carrión & Casacuberta, 2024). The motivation behind introducing an interactive approach is that, in many real-world scenarios, human users or domain experts need a fine-grained control over the model's domain expertise. In such interactive environments, quick adjustments of domain knowledge are crucial to accommodate these evolving requirements. Therefore, by allowing users to calibrate or boost the domain expertise more easily during inference, we can quickly adapt to evolving environments without fine-tuning the entire model.

To achieve this, rather than training separate low-rank matrices for each task and switching between them on demand, we propose an approach inspired by a Mixture of Experts (MoE), using a linear combination of LoRA adapters, termed Mixture of LoRA Experts (MoLE) for simplicity here (Carrión & Casacuberta, 2024). That is, we use a calibrated linear combination of compatible low-rank matrices trained across different domains and styles, but without employing a gating network. Thus, enabling on-the-fly adjustments to bias the model's knowledge towards desired domains or styles, without the need for additional fine-tuning.

Unlike traditional mixtures of experts that rely on networks to weigh the contributions of each expert, this approach allowed us to use domain-specific coefficients to calibrate the influence of each LoRA matrix avoiding any single domain dominance. Besides, a user-specific scaling factor was added to improve the interactivity control, allowing further customization of the domain contributions. The adapted model weights are calculated as follows:

$$W'_{\text{MoLE}} = W + \sum_{n=1}^{K} \alpha_n \lambda_n X_n Y_n^{\top} \tag{6.9}$$

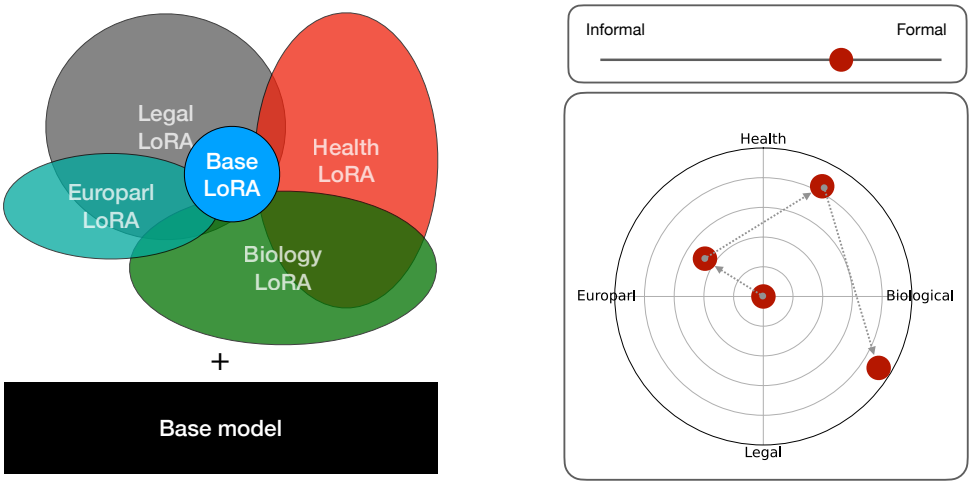
where W represents the base model weights, K denotes the number of domains or styles, X_n, Y_n are the learned low-rank adapter matrices[6] associated with each task $\mathcal{T}_n$ (here, domain), λ_n are domain-specific coefficients that balance the individual contributions from each domain, and α_n is a user scaling factor that the user can adjust to increase or decrease the influence of a given domain's adapter on the final output. This formulation ensures a balanced performance across multiple domains, preventing any single domain from overly influencing the model performance in one direction.

To implement this strategy, we first pre-trained a base NMT model on a diverse set of datasets covering various domains to build a robust and generic translation model. Next, instead of training individual low-rank matrices from scratch for each domain, we pre-trained a base LoRA on the same multi-domain data to get a low-rank representation fully aligned with the base model. Then, we used this generic LoRA as a foundational adapter to be fine-tuned on specialized domains while preserving an overlap in the weight distributions between the adapters and the base models. This was done to facilitate a smoother domain transition (see Figure 6.12a).

To facilitate a more intuitively user interaction, we draw inspiration from the Interactive Machine Translation (IMT) paradigm discussed in Section 2.6.1, where human feedback can guide model outputs at inference time. Here, we adapt that concept to the domain adaptation problem so that users can nuance the domain expertise (or style) of a model interactively if the current translation fails to meet their expectations. Initially, we developed a simple text-based interface that allowed users to adjust the domain-specific coefficients λ_n. However, for a more intuitive user experience, Figure 6.12b depicts a Graphical User Interface (GUI), where users can easily control each domain-specific weight contribution via sliders or input fields, making domain and style adaptation more accessible and user-friendly.

To evaluate the effectiveness of this strategy, we conducted several evaluations to measure both, the absolute and relative performance improvements across all these domains. As illustrated in Figure 6.13, this approach allowed us to interactively adjust the model's performance for each domain continually

[6]Note that these low-rank components, X_n, Y_n, are not the source-target pairs (x, y) used for training, but rather parameter matrices for adaptation.



(a) Diagram of MoLE architecture.

(b) GUI for controlling our MoLE.

Figure 6.12: Interactive Adaptation Components: On the left, the diagram illustrating the MoLE architecture used for parameter-efficient adaptation in NMT, and the conceptual domain overlapping. On the right, the proposed GUI for controlling domain-specific contributions, thus facilitating user-driven domain adaptations.

by modifying the aforementioned coefficients λ_n. Accordingly, Figure 6.13(a) shows the absolute BLEU scores for the base model along with the enhanced performance area achieved through this interactive approach across the various domains. In contrast, Figure 6.13(b) illustrates the relative performance gains as a function of the rank, highlighting the relation between the parameter count and performance improvements.

From our experimentation, we noticed that the MoLE often behaves as an ensemble of specialized models, which usually leads to performance gains across all domains rather than just on those with the increased coefficients. However, we also identified several limitations with this method. First, the influence of each low-rank matrix on the final model performance was not always directly proportional to their weighting coefficient, as some domains contributed more strongly than others. This happened despite the calibration and normalization efforts, given that the interactions between the different adapters became increasingly non-linear and unpredictable as domain differences grew. Second, this approach did not address the CF problem directly. Indeed, improving the performance in one domain by increasing its weight contribution sometimes

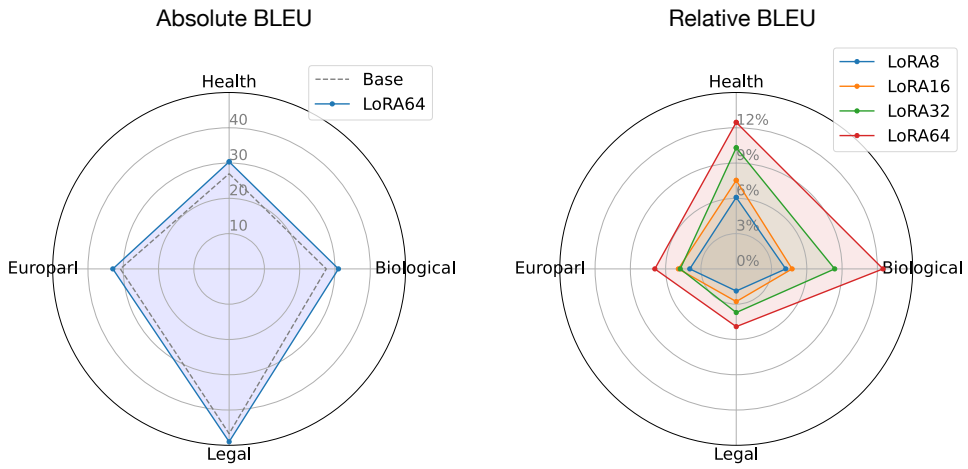


Figure 6.13: Interactive domain adaptation results: (a) Absolute BLEU scores for the base model (dashed line) and performance improvements (shaded area) across four domains (health, biological, legal, and europarliamentary) after applying the interactive low-rank strategy. (b) Relative performance gains (%) at different ranks (8, 16, 32, 64).

led to a degradation in the performance of the other domains, especially when these domains were substantially different (e.g., Legal and Heath).

As a result, this interactive domain adaptation strategy, based on a calibrated MoLE, offers a simple, flexible, and efficient mechanism to adapt the style and domain expertise of NMT models to new domains without the need for fine-tuning. However, the non-linear interplay between adapters, in addition to the catastrophic forgetting effects, suggests a need for further regularization mechanisms that will be discussed in Section 6.4.3.

6.4.3 Gradient-Based Regularization for Knowledge Integration in Low-Rank Matrices

To address the CF problem while integrating new knowledge into these low-rank matrices, we adhere to the model space hypothesis discussed in Section 6.1.1. There, we hypothesized that a pre-trained model f_θ could be viewed as a point within a high-dimensional space, where each task defines a region in that feature space. Consequently, the problem of mitigating the catastrophic forgetting phenomenon becomes the problem of moving our NMT model (point) into the overlapping region shared by both tasks (assuming such an overlap exists).

Then, in order to move beyond the rehearsal strategies discussed earlier, we considered a scenario where we had access only to the weights of the previous model and the data for the new task, which is a fairly common situation in practical lifelong learning. This limitation prevented us from simply merging the old and the new datasets, which would have allowed us to identify the shared region between tasks more easily. Instead, we adopted an algebraic approach that penalized weight deviations, in direction and magnitude, relative to the previous model.

Building on this idea, our goal for this study was to adapt a given pre-trained model f_θ on task $\mathcal{T}_A$, thus f_{θ_A}, to a new task $\mathcal{T}_B$, while minimizing the performance degradation on the original task $\mathcal{T}_A$. Hence, resulting in a new model $f_{\theta_{A,B}}$. To simplify the notation, we can represent these models (or hyperdimensional points) as high-dimensional parameter vectors such as θ_A, θ_B, or $\theta_{A,B}$, where θ_A represents the parameters of the model f_θ trained on task $\mathcal{T}_A$, θ_B represents the parameters of the model f_θ trained on task $\mathcal{T}_B$, and $\theta_{A,B}$ represents the parameters of the model f_θ trained on both tasks $\mathcal{T}_A$ and $\mathcal{T}_B$ (merged). Therefore, the transformation of the vector θ_A into $\theta_{A,B}$ could be viewed as adding the difference $\theta_{A,B} - \theta_A$ to the original vector θ_A.

However, since the exact location of $\theta_{A,B}$ is unknown, we need to introduce a penalty term in our objective function to control the magnitude (distance) of these parameter updates, with the goal of preventing $\theta_{A,B}$ from moving too far from θ_A. Specifically, we use the norm of their difference raised to a tunable power, $\|\theta_{A,B} - \theta_A\|^\gamma$, allowing finer control over the regularization effect. Besides, since the direction in which $\theta_{A,B}$ should move towards θ_B is determined by the new task $\mathcal{T}_B$, moving directly toward θ_B without any constraints might lead us out of the desired overlapping region of the tasks (see Figure 6.1). Therefore, to mitigate this effect, we had to adjust the new parameter updates by considering the importance of parameters from the previous task, similar to pruning strategies (Franchi et al., 2019; Z. Zhang et al., 2023). To do so, we calculate this importance using the cumulative gradient of the previous model f_A after training it on task $\mathcal{T}_A$, as follows:

$$G_\theta = \frac{1}{M} \sum_{i=1}^{M} \nabla \mathcal{L}(f_\theta, x^{(i)}, y^{(i)}) \tag{6.10}$$

where G_θ is the average accumulated gradient over all these training examples[7], M represents the total number of training samples from the corresponding previous task (e.g., $\mathcal{T}_A$), and $\nabla\mathcal{L}(f_\theta, x^{(i)}, y^{(i)})$ represents the gradient of the loss function $\mathcal{L}$ with respect to the model parameters θ for a single training example $(x^{(i)}, y^{(i)})$.

Accordingly, by combining all these components into a single loss function derived from a standard loss, we created a gradient-based regularization strategy specifically designed for our low-rank matrices:

$$\mathcal{L}'_K = \mathcal{L}(X_K, Y_K) + \lambda_{reg} \sum_{n=1}^{K-1} \Big[G_{X,n} \big\| (X - X_n) \big\|^\gamma \; + \; G_{Y,n} \big\| (Y - Y_n) \big\|^\gamma \Big] \quad (6.11)$$

where $\mathcal{L}$ is the primary task loss (e.g., cross-entropy) for the current task $\mathcal{T}_K$, X_n and Y_n are the low-rank matrices learned for task $\mathcal{T}_n$, λ_{reg} is a regularization coefficient that controls the influence of the penalty term, $G_{X,n}$ and $G_{Y,n}$ represent the cumulative gradients (or importance measures) for task $\mathcal{T}_n$ with respect to the corresponding parameters X and Y, γ is an exponent that controls the type of norm used[8].

Consequently, the optimization objective for these regularized low-rank matrices tries to identify the optimal low-rank matrices X_K^* and Y_K^* that minimize the previously defined loss function $\mathcal{L}'_K$. Hence, we define our optimization objective as follows:

$$(X_K^*, Y_K^*) = \arg\min_{X_K, Y_K} \mathcal{L}'_K(X_K, Y_K; X_{1,\ldots,K-1}, Y_{1,\ldots,K-1}, G_{X,1,\ldots,K-1}, G_{Y,1,\ldots,K-1}) \quad (6.12)$$

Through this mathematical formulation shown in equations 6.11 and 6.12, we seek to mitigate the CF problem in a parameter-efficient manner using low-rank matrices, to enable an efficient integration of new knowledge into these matrices.

Accordingly, in this section, we study this strategy under two challenging scenarios:

[7] In practice, this can be computed over a full dataset or a representative subset, making it effectively an approximation of the true expected gradient.

[8] If $\gamma = 1$, the regularization behaves like an L1 norm. And if $\gamma = 2$, the regularization behaves like an L2 norm.

- First, we try to incorporate a new domain knowledge (i.e., legal) into a set of pre-trained adapters for a health domain while keeping the language pair fixed (see Section 6.4.3).
- Second, we tackle a more challenging scenario, where we try to incorporate a new language pair (i.e., English-French) into an adapter that uses a different language pair (i.e., English-Spanish), again, without forgetting the original language pair and without access to any past data samples during the training process (see Figure 6.15).

Incorporating new domains into pre-trained low-rank adapters

In this first scenario, we attempted to incorporate new domain expertise (i.e., legal) into a set of pre-trained adapters for the health domain while keeping the language-pair fixed (Carrión & Casacuberta, 2024). Again, this was done without access to past data from that domain (i.e., health).

As a result, in Figure 6.14, we show the BLEU performance of a set of NMT models evaluated on both the old and new domains (Health and Legal) during the learning process of the new domain (legal). It is important to point out that, to better capture the learning dynamics of each approach, Figure 6.14 reports BLEU scores on the validation set during training, as this was preferred rather than relying on a single test set evaluation data point which could be more susceptible to noise. Moreover, since both the validation and test sets were randomly sampled from the same dataset, the BLEU performance was remarkably similar on both sets. Accordingly, the base model was initially trained on the health domain as a starting point, and then, we fine-tuned it on the legal domain (without access to the previous data). To do so, we employed the gradient-based regularization for the low-rank matrices described in the previous section (see Section 6.4.3). For comparison, we also included models trained without regularization (red lines) and with L2 regularization (blue lines).

As shown in Figure 6.14, there is a clear trade-off between learning the new domain and retaining knowledge from the old domain. Our gradient-based regularization strategy (solid green line) demonstrates a strong resilience to the effects of the catastrophic forgetting phenomenon during the learning of the new legal domain. However, it also shows some difficulty in learning the new task due to the added complexity of preserving previous knowledge while integrating new information in a highly compressed space (low-rank matrices). The L2 regularization approach (dashed blue line) provides less control over the

Figure 6.14: CF regularization strategies in new domain scenarios: The left figure shows the forgetting of the previous domain (health), while the right figure shows the learning of the new domain (legal). Our gradient-based regularization strategy (solid green line) showed the strongest resilience to catastrophic forgetting among the tested methods, despite a slight reduction in the rate of learning for the new domain. The L2 regularization (dashed blue line) serves as a middle ground, while the non-regularized approach (dashed red line) learns the new domain quickly but forgets the previous domain entirely (preserving some performance due to the common language pair). (Note: *run_id* 's values are used to match training runs between both plots).

catastrophic forgetting but allows for a more efficient learning of the new domain, acting as an intermediate solution between our method and the non-regularized approach. Then, the non-regularized model (dashed red line) learns the new task very efficiently, as there are no learning constraints, but it practically forgets all the previous domain expertise (it retained a decent performance due to the common language pair).

Therefore, given that our method offers a high control over the plasticity-elasticity dilemma, we argue that by adjusting the given hyperparameters, we can achieve a satisfactory balance between the learning rate for which the new task is learned, while significantly mitigating the catastrophic forgetting effects on the previous task.

Incorporating new languages into pre-trained low-rank adapters

In the second and more challenging scenario, we attempted to add a new language pair (i.e., English-French) into a low-rank adapter that used a different language pair (i.e., English-Spanish), again without access to past training data during the learning of the new language pair (see Figure 6.15) (Carrión & Casacuberta, 2024).

To do so, we began with a pre-trained model on an English-Spanish dataset with an adapter. Then, we applied our gradient-based regularization strategy to incorporate the new language pair (English-French) into this pre-trained low-rank adapter that used English-Spanish as its only language pair. This was an extremely difficult challenge because we had to compress two language pairs in a very small space.

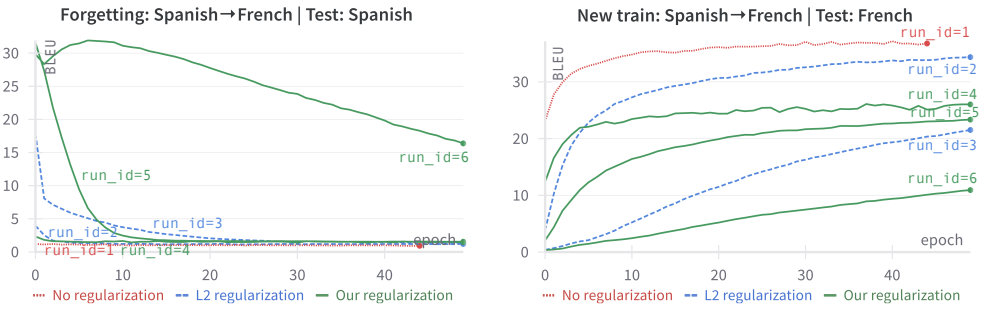


Figure 6.15: CF regularization strategies in new language-pair scenarios: The left figure shows the forgetting of the previous language pair (English-Spanish), while the right figure shows the learning of the new language pair (English-French). Although none of the approaches were entirely successful in balancing the forgetting and the learning of these two language pairs in the limited space due to the task's complexity, our gradient-based regularization (solid green line) was the only method that could retain enough of the past knowledge while the new language-pair was being learned (see *run_id=6*). In contrast, the other approaches resulted in an immediate collapse of the previous knowledge during the first training iterations.

This task proved to be particularly difficult for all strategies evaluated (see Figure 6.15), as none of the tested methods were able to successfully incorporate the second language pair from scratch using low-rank matrices while retaining most of the previous knowledge. We argue that the difficulty in accommodating both language pairs within the low-rank matrices (on top of a monolingual NMT model) is due to network saturation issues, where the limited parameter capacity creates an artificial constraint. In addition to this, we could have tried

larger low-rank matrices and a larger model, but due to hardware limitations, we could not conduct such experiments.

Therefore, our gradient-based regularization strategy was the only strategy that did not result in an immediate collapse of performance on the previous language pair (English-Spanish) during the initial training iterations of the new language pair (see the right figure of Figure 6.15, *run_id=6*). Instead, both the forgetting and learning curves were smooth and gradual. Despite this, the task remains highly challenging and requires further investigation in future work to push its boundaries. However, we believe that with better hardware, larger base models, and more parameters in our adapters, this technique could be as feasible for incorporating new language pairs as it proved to be the case when incorporating a new domain (see Section 6.4.3).

6.5 Discussion and Conclusions

In this chapter, we addressed the fundamental challenge of the Catastrophic Forgetting in NMT by examining how models can be continually adapted without losing any of their previously acquired knowledge. Our research approached this problem from multiple perspectives.

First, we investigated the vocabulary's influence on the CF problem, showing that a mismatch between the vocabulary domain and the target domain can significantly exacerbate forgetting. We observed that a more general and continual vocabulary, built on compositional embeddings, can effectively alleviate CF. However, despite the modest initial gains, these approaches showed a significant cross-domain consistency while enabling new words to be added on the fly without disrupting the embedding space.

Second, we evaluated strategies to mitigate the CF through rehearsal and regularization. Our experiments on multilingual NMT models revealed that by interleaving even a very small percentage (1%) of past data during a sequential training, we can significantly preserve most of the previously acquired knowledge. Furthermore, we showed that by using a simple few-shot rehearsal mechanism, combined with a re-weighting strategy and a customized loss regularization based on a knowledge distillation technique to penalize weights that deviate too much from the original ones, we can drastically limit the weight updates that are susceptible of overwriting past relevant knowledge, thus striking a balance between stability and plasticity dilemma.

Lastly, we studied parameter-efficient adaptations, particularly low-rank adaptation methods such as LoRA. Through these methods, we showed that a minimal set of additional parameters can be used to decouple the shared components of our models from task-specific ones, enabling models to adapt to multiple domains and styles with a minimal storage overhead. However, while this approach circumvents CF by decoupling the base translation functionality from task-specific adaptations, it has some limitations: slower convergence compared to traditional fine-tuning, restricted expressiveness at very low ranks, and a notable risk of saturation when compressing multiple complex tasks (e.g., languages) into a single low-rank space. Nonetheless, these techniques remain highly promising for scenarios where memory constraints or efficient task-switching are important.

In conclusion, our findings reveal that no single strategy can completely solve the CF problem on its own. Instead, each approach addresses a specific aspect of this challenge. For instance, continual vocabularies help to effectively maintain a granular lexical coverage across unseen words, domains, and even new languages. Rehearsal and regularization mechanisms allow us to dramatically reduce the catastrophic forgetting, even when access to past data is very limited. Meanwhile, parameter-efficient adaptations enable efficient task-switching by isolating task-specific knowledge from the core semantic representations of our models. Collectively, these strategies provide a robust framework for NMT systems capable of evolving over time, adapting to new linguistic scenarios while preserving most of the previously acquired knowledge. Therefore, as research progresses, future innovations will appear, likely unifying these strategies into a cohesive, end-to-end model perfectly suited for the dynamic and ever-changing linguistic landscapes of our world.

Chapter 7

Conclusions

This chapter summarizes the key findings and contributions of this thesis, which focuses on addressing two significant challenges in Neural Machine Translation (NMT): The open-vocabulary problem and the catastrophic forgetting phenomenon. We presented novel approaches, including Quasi-Character-level vocabularies, parameter-efficient adaptation strategies, and several regularization strategies, to improve the model's ability to learn from continual data scenarios without significantly degrading the performance on previously learned tasks. The implications of these findings for Continual Learning (CL) in NMT are discussed, along with the limitations of the current work and suggestions for future research directions.

7.1 Summary of Research

The primary objective of this work was to advance the field of Continual Learning in NMT by addressing the open-vocabulary problem and the catastrophic forgetting phenomenon. To achieve this, we explored various vocabulary encoding strategies and developed methods to enable NMT models to continuously adapt to new linguistic scenarios without extensive retraining or significant performance losses in previously learned tasks. Our core contributions are found in Chapter 5 and Chapter 6.

In **Chapter 5**, we investigated the open-vocabulary problem, which arises due to the ever-evolving nature of human language and the limitations of fixed-

vocabulary NMT models in handling Out-of-Vocabulary (OOV) words. We analyzed the impact of different vocabulary encodings, proposed a novel Quasi-Character-level encoding, and explored incremental and continual vocabularies through a compositional embedding strategy.

In **Chapter 6**, we focused on the catastrophic forgetting problem in NMT, where models tend to forget previously learned information upon learning new one. We examined the influence of vocabulary domain and size on the model's retention capabilities, studied rehearsal and regularization to mitigate these effects using minimal information, explored the effectiveness of parameter-efficient adaptation methods for circumventing this problem, and proposed a gradient-based regularization approach to incorporate new knowledge into these highly-compressed low-rank adapters.

7.2 Key Findings and Contributions

This section summarizes the key findings and contributions of our research, emphasizing the proposed approaches to tackle the open-vocabulary problem and mitigating the catastrophic forgetting in NMT.

7.2.1 Addressing the Open-Vocabulary Problem in NMT

To address the open-vocabulary problem, we explored novel vocabulary strategies that balance sequence length, information density, and compositionality.

Vocabulary encoding strategies

We conducted a comprehensive analysis of various vocabulary encoding strategies, including byte-level, character-level, subword-level, and word-level encodings. Our general findings indicated that:

- **No vocabulary is inherently superior in theory:** From a grammatical equivalence standpoint, different encodings can represent the same linguistic information. However, practical considerations such as handling unknown tokens, compositionality, and computational efficiency significantly impact their effectiveness.
- **Trade-offs in information density and sequence length:** Low-level encodings result in longer sequences with low-information density tokens, which pose significant challenges in learning long-term dependencies. In

contrast, high-level encodings have high-information density tokens but struggle with OOV words due to limited compositionality.

Quasi-character-level vocabularies

We introduced quasi-character-level vocabularies as a novel encoding strategy that extends character-based vocabularies by incorporating the most frequent character pairs (Carrión & Casacuberta, 2021, 2022c). The key benefits of this approach include:

- **Exponential reduction in tokens:** By combining single characters with the most frequent character pairs, we achieved a significant reduction in sequence length compared to pure character-level encodings, improving computational efficiency.
- **Improved learning efficiency:** Quasi-character models exhibited higher token utilization due to shorter long-tail distributions, enhancing learning efficiency, particularly in low-resource scenarios.
- **Facilitating learning of long-term dependencies:** Shorter sequences resulting from Quasi-character vocabularies enabled models to capture long-term dependencies more effectively, improving translation quality.
- **Reducing memory and computational requirements:** The reduced sequence lengths alleviated the computational burden associated with the quadratic processing of long sequences in NMT models, optimizing resource utilization.

Our experimentation has demonstrated that Quasi-character-level vocabularies outperformed other encoding strategies in low- to mid-resource scenarios and maintained a competitive performance in high-resource scenarios. Moreover, the advantages of this approach were consistent across different languages, domains, and neural architectures.

Incremental and continual vocabularies with compositional embeddings

To address the need for integrating new vocabulary terms without extensive retraining, we explored the following ideas:

- **Incremental vocabularies:** We assessed zero-shot translation capabilities at the vocabulary level (i.e., new semantics), emphasizing the

importance of dimensionality matching, embedding quality, and embedding alignment when introducing new word embeddings into pre-trained models (Carrión & Casacuberta, 2022a).

- **Continual vocabularies with compositional embeddings:** By leveraging compositional subword information, we enabled models to generate embeddings for unseen words dynamically, reducing the need for vocabulary expansion or fine-tuning unless entirely new semantics were introduced (Carrión & Casacuberta, 2023b).

These approaches allowed NMT models to adapt to new linguistic terms efficiently while maintaining translation quality over time.

7.2.2 Mitigating the Catastrophic Forgetting in NMT

To mitigate the CF problem, we investigated the vocabulary influence on the model's retention capabilities and explored solutions including few-shot rehearsal strategies, advanced regularization techniques, and parameter-efficient adaptation mechanisms.

Vocabulary influence on the catastrophic forgetting

Our investigation into the impact of vocabulary domain and size on model retention (Carrión & Casacuberta, 2023b), revealed that:

- **Alignment of vocabulary and test domains:** Models performed best when the vocabulary domain matched the test domain, thus emphasizing the importance of vocabulary selection in mitigating catastrophic forgetting.
- **Use of continual vocabularies:** Implementing a universal and continual vocabulary with compositional embeddings improves the performance consistency (i.e., standard deviation) across domains and mitigates the effects of catastrophic forgetting compared to domain-specific vocabularies.

These findings highlight the role of vocabulary strategies in enhancing cross-domain generalization and knowledge retention in NMT models.

Rehearsal and regularization strategies

We also explored methods to mitigate the catastrophic forgetting problem by incorporating minimal rehearsal mechanisms and regularization techniques (Carrión & Casacuberta, 2022b):

- **Minimal rehearsal:** Incorporating tiny amounts of past data during the learning of new tasks significantly reduced catastrophic forgetting, even when using as little as 1% of past data per batch.
- **Few-shot rehearsal with oversampling and regularization:** Combining a minimal rehearsal strategy with oversampling and a customized focal loss function that penalizes weight deviations from the original model, we further improved knowledge retention while integrating new information efficiently.

These strategies demonstrated a practical and simple approach to maintain the model performance on previous tasks without an excessive reliance on past data. However, these strategies also suffer from network saturation problems, so the number of tasks that a network can learn is limited by the complexity of the task and the parameter count of the model.

Parameter-efficient adaptation

We introduced a parameter-efficient adaptation approach based on the Low-Rank Adaptation (LoRA) technique to decouple task-specific components from the shared semantic structures of our models (Carrión & Casacuberta, 2024):

- **Efficient task-switching:** Low-Rank adapters enabled efficient adaptation of NMT models to new domains, styles, and languages using only a minimal fraction of the model's parameters, significantly reducing the computational costs.
- **Interactive domain adaptation:** A simple Mixture of LoRA Experts (MoLE) approach allowed for an interactive adaptation of these models to different domains and styles without any additional fine-tuning, offering flexibility and user-friendly control.
- **Gradient-based regularization for low-rank matrices:** We developed a gradient-based regularization strategy specifically designed for low-rank matrices to mitigate catastrophic forgetting while integrating new knowledge into these high-information dense components.

These methods provided scalable solutions for practical continual learning in NMT, circumventing or balancing the stability-plasticity dilemma inherent in integrating new knowledge.

7.3 Implications for Continual Learning in NMT

The contributions of this work have significant implications for advancing Continual Learning in NMT:

- **Cross-domain generalization:** The use of continual vocabularies and compositional embeddings enhances the models' ability to generalize across different domains, improving robustness and versatility.
- **Enhanced adaptability:** The proposed continual vocabulary strategies and parameter-efficient adaptation methods enable NMT models to adapt continuously to new linguistic scenarios, including new words, domains, styles, and languages, without significantly sacrificing performance on previously learned tasks.
- **Improved knowledge retention:** The minimal rehearsal and regularization strategies effectively mitigate catastrophic forgetting, ensuring that models retain valuable knowledge over time, which is essential for maintaining translation quality in dynamic linguistic landscapes.
- **Resource optimization:** By leveraging efficient encoding strategies and low-rank adaptations, these approaches minimize the computational requirements, enabling scalability and practical deployment of lifelong NMT systems, especially in resource-constrained environments.

Collectively, these findings contribute to the development of more resilient and adaptive NMT systems, capable of meeting the demands of the ever-evolving human languages and user needs.

7.4 Limitations

While the approaches presented in this work offer significant advancements, several limitations still need careful consideration:

- **Non-linear interactions in adapter combinations:** In the interactive domain adaptation approach, the non-linear interplay between different

adapters sometimes led to unpredictable performance shifts, indicating the need for better and more sophisticated mixture strategies.

- **Complex task integration in low-rank adaptations:** Incorporating entirely new languages or highly complex tasks into pre-trained models using low-rank adaptations remains quite challenging with common hardware due to the limited parameter space and network saturation problems, which leads to performance trade-offs.
- **Control over the catastrophic forgetting:** Although the gradient-based regularization strategy provides better control over the stability-plasticity dilemma (especially for styles and domains), it still requires careful consideration of complex tasks (i.e., languages).
- **Computational constraints:** Some experiments were limited by our hardware resources, restricting the exploration of larger models or higher-rank adaptations that might overcome some of the challenges observed here.

Acknowledging these limitations provides better insights for refining these contributions and serves as a guide for future research efforts.

7.5 Future Work

Building upon the findings and recognizing their limitations, several paths for future research are suggested:

- **Scaling up experiments:** Utilize more hardware resources to explore the effects of larger models and higher-rank adaptations in order to provide deeper insights into the scalability and performance limits of the proposed methods.
- **Adaptive regularization techniques:** Develop more sophisticated regularization methods that can be dynamically adjusted based on the task complexity or on the network saturation levels.
- **Advanced adapter combination strategies:** Explore non-linear combination methods and gating mechanisms in the interactive adaptation strategy to better control the interplay between multiple adapters and improve the performance consistency.

- **Better low-rank adaptations for new language pairs:** Investigate strategies to increase the parameter sharing or the compression levels of low-rank adapters to improve the integration of entirely new language pairs.
- **Integration with Complementary Learning Systems (CLS):** Investigate the integration of CLS that combine fast adaptation with stable long-term retention to improve the overall continual learning capabilities of NMT models.
- **Extending this work to other Natural Language Processing tasks:** Evaluate the applicability of the proposed strategies to other natural language processing tasks, such as language modeling, text summarization, or question answering.

We believe that pursuing these research directions will help to overcome current challenges and significantly advance the field of Continual Learning in NMT and related areas.

7.6 Final Reflections on the Fast Pace of AI Research

Beyond the specific contributions to Continual Learning in NMT, it is important to reflect on the broader context in which this work has been developed. The period between 2023 and 2025 has been characterized by an unprecedented acceleration in the capabilities of Artificial Intelligence (AI), particularly in Large Language Models (LLMs). OpenAI's GPT-4 was soon followed by GPT-4o, a multimodal model capable of processing text, images, and audio in real time (OpenAI et al., 2024). Later that year, they introduced the o1 series, models specifically designed for complex reasoning tasks by *thinking* longer before responding. Anthropic released Claude 3.5 and Claude 3.7 (Anthropic, 2024a), both optimized for reasoning and long-context understanding. Google's Gemini 2.5 Pro and Meta's LLaMA 4 (Google DeepMind, 2025; Meta AI, 2025) further contributed to the growing diversity of high-performing models, with open benchmarks like Chatbot Arena ranking them competitively across multilingual, multitask, and interactive settings (Chiang et al., 2024).

In this rapidly evolving context, foundational tasks such as NMT are being redefined. Frontier models increasingly exhibit few-shot or even zero-shot translation capabilities, which raises the performance baseline but does not negate the need for specialized solutions. Particularly in dynamic or domain-sensitive settings, the ability to continuously adapt without retraining remains

a core requirement. The pace of change in AI thus reinforces the motivations behind this thesis: developing systems that learn continually and remain effective over time.

Another notable trend is the growing distinction between models that rely on pattern matching and those that demonstrate structured reasoning. While LLMs are still grounded in statistical learning, techniques such as Chain-of-Thought (CoT) prompting, Tree-of-Thought (ToT), tool use, and test-time search have significantly enhanced their ability to solve reasoning-intensive tasks (Bi et al., 2025; Yang et al., 2025). OpenAI's o3 model, for instance, achieved human-level performance on ARC-AGI, a benchmark once thought unreachable for LLMs (Dickson, 2024; Jones, 2025). This shift toward reasoning-centric AI has implications for continual learning, where fast test-time adaptation can complement slower, weight-level updates.

In parallel, Retrieval-Augmented Generation (RAG) has emerged as a prominent method to address the knowledge cutoff problem (P. Lewis et al., 2020). By dynamically incorporating external information at inference time, RAG systems reduce the frequency and cost of retraining. While the techniques explored in this thesis focus on internal adaptation (through rehearsal, regularization, and low-rank updates), RAG offers a complementary external strategy. Future work may benefit from hybrid models that combine these internal and external mechanisms for lifelong learning.

Evaluation methodologies are also evolving to reflect these trends. Metrics like Bilingual Evaluation Understudy (BLEU) continue to serve a role, but benchmarks such as MMLU Pro, GPQA Diamond, MATH-500, ARC-AGI and Chatbot Arena provide deeper insights into reasoning, generalization, and human preferences (Chiang et al., 2024; Chollet, 2019; Hendrycks et al., 2021; Rein et al., 2023; Y. Wang et al., 2024). These benchmarks challenge models to solve novel, non-memorizable problems, an increasingly important proxy for general learning ability in dynamic environments.

Moreover, the growing emphasis on Agentic AI, where systems orchestrate tools, retrieve relevant information, and interact with modular components, highlights the importance of scalable adaptation. Frameworks like HuggingGPT exemplify this modularity, enabling AI models to integrate various tools and data sources seamlessly through modular pipelines (Shen et al., 2023). Similarly, Anthropic's Model Context Protocol (MCP) proposes a standardized interface for connecting AI systems with external data sources and tools, allowing dynamic, context-aware workflows without modifying the underlying model (Anthropic, 2024b). In such architectures, continual learning methods that

enable selective updating or modular extension can offer efficient paths towards lifelong adaptation without the burden of retraining entire systems.

Finally, these developments unfold amid broader debates on the trajectory towards Artificial General Intelligence (AGI). While some people, like Ray Kurzweil, forecast human-level AI by 2029 (Kurzweil, 2024), others stress the architectural and alignment challenges that remain (Amodei & Hernandez, 2018; LeCun, 2022). Analysts such as Aschenbrenner (Aschenbrenner, 2024) suggest that AI may soon automate its own improvement processes, rapidly accelerating progress towards Artificial Superintelligence (ASI).

Against this backdrop, the relevance of this thesis becomes even more pronounced. The challenges of catastrophic forgetting, vocabulary adaptation, and efficient lifelong learning are unlikely to be solved by scale alone. As AI systems become more complex and modular, the need for robust and interpretable continual learning strategies will only grow. Whether through internal updates, retrieval-based adaptation, or orchestration frameworks, the ability to remain effective in changing environments is becoming a defining characteristic of modern AI.

In summary, the fast pace of AI research reaffirms the necessity of building systems that adapt continuously. This work contributes to that goal by proposing methods that are not only effective in the current landscape but are also designed to remain relevant as that landscape evolves. We hope that these contributions will serve as a foundation for further research in lifelong learning and the next generation of adaptable, intelligent systems.

Appendix A

Supporting Frameworks

A.1 AutoNMT

AutoNMT is a Python framework created to streamline the research of sequence-to-sequence models, focusing primarily on Neural Machine Translation (NMT). The complexities inherent in the Machine Learning (ML) experimentation, such as data pre-processing, model configuration, evaluation, and standardized reporting, are addressed through an extensive pipeline provided by this framework (Carrión & Casacuberta, 2023a), whose goal is to reduce the manual overhead in experimentation, allowing researchers to focus more on model innovation and analysis.

At the core of AutoNMT is the *DatasetBuilder* component, which handles data management and pre-processing. This component enables researchers to generate multiple dataset variations using customizable factors like normalization, tokenizations, dataset length, and vocabulary size. By automating these variations, the framework allows a systematic exploration of the effects of different configurations on the model performance and generalization.

Complementing the *DatasetBuilder* is the *Meta-Trainer*, a toolkit-agnostic training interface to support popular sequence-to-sequence frameworks such as Fairseq (Ott et al., 2019), OpenNMT (Klein et al., 2017), and AutoNMT's

internal toolkit. Through this interface, the Meta-Trainer automates both, the training and the evaluation process, facilitating controlled comparisons across models trained under different conditions and datasets. This automation not only standardizes the experimentation part, but also enhances the flexibility, enabling researchers to incorporate custom model designs, tokenization methods, and evaluation metrics. By addressing typical challenges in Natural Language Processing (NLP) research, AutoNMT has proven to be an highly reliable tool for studying the Continual Learning in NMT.

A.2 DeepHealth Toolkit

The DeepHealth Toolkit, funded by the European Union's Horizon 2020 initiative, is an open-source deep learning framework specifically developed to enhance productivity in the biomedical field through distributed and High-Performance Computing (HPC) solutions (Cancilla et al., 2021). The toolkit comprises two main libraries, European Distributed Deep Learning Library (EDDL) and European Computer Vision Library (ECVL), which collectively enable a seamless workflow for computer vision and Deep Learning (DL) tasks. These libraries are designed with hardware transparency from the ground up, enabling them to operate across CPUs, GPUs, and FPGAs. This approach allows the toolkit to support the deployment of deep learning models within HPC and cloud environments, optimizing the resource utilization and reducing the computational bottlenecks.

The EDDL offers extensive support for tensor operations and distributed deep learning, including both training and inference, with optimizations tailored for biomedical applications. It enables multi-hardware compatibility, allowing models to be trained on heterogeneous architectures and transparently scaled across HPC or cloud resources. Complementing EDDL, the ECVL advanced data processing and augmentation capabilities, which are essential for managing and preparing biomedical images, among other data types. Together, EDDL and ECVL create an integrated, end-to-end workflow, from data pre-processing to model deployment, that simplifies complex tasks like image segmentation and classification in large biomedical datasets.

In the context of continual learning, DeepHealth's architecture was instrumental in shaping the design of the AutoNMT framework. Its distributed training capabilities, modularity, and emphasis on user-friendliness and productivity served as guiding principles in developing this sequence-to-sequence framework,

reflecting the DeepHealth's impact on both the technical and philosophical aspects of our continual learning research.

A.3 SELENE

The SELENE project is focused on delivering a high-performance, RISC-V-based System-on-Chip (SoC) optimized for real-time neural network inference in safety-critical applications, such as autonomous systems and robotics (Medina et al., 2022). Built with efficiency and reliability as top priorities, SELENE meets the rigorous demands of these domains by incorporating six NOEL-V RISC-V cores and specialized Artificial Intelligence (AI) accelerators, which enhance both computational speed and power efficiency in neural network tasks.

A pivotal feature of SELENE is its SELENE Acceleration Framework (SAF), which acts as an interface between hardware accelerators and software layers, particularly the EDDL. The SAF enables seamless integration of neural models with SELENE's high-performance hardware without the need for extensive software modifications. By supporting FPGA and ASIC implementations, the framework delivers the flexibility to deploy and scale across various real-time applications with significant hardware optimizations. This high-speed AXI interconnect, alongside advanced DMA capabilities, ensures efficient data flow across the system, which is crucial for applications with tight latency requirements, such as robotics and automotive control systems.

The EDDL integration within SELENE facilitates the continual learning in a real-time environment by allowing models to be updated and retrained directly on hardware. This setup is particularly beneficial for applications requiring continual adaptations, as it enables the system to incrementally learn from new data without the need for complete retraining, reducing both the computation time and the memory usage. Furthermore, SELENE's architecture and SAF's compatibility with distributed learning were influential in shaping the design of AutoNMT, particularly in how it supports modular, toolkit-agnostic training on heterogeneous hardware.

Appendix B

List of Publications

B.1 Publications Directly Supporting this Thesis

This thesis builds upon the following publications, which reflect the evolution of the PhD research and provide support to the findings presented throughout the manuscript. For clarity, the publications are grouped according to their main research focus and the chapter in which their contributions are discussed.

B.1.1 Addressing the Open Vocabulary Problem in NMT

The following works address the open vocabulary problem by exploring different encoding strategies for handling unseen or rare words, with a particular focus on low-resource scenarios and the continuously evolving nature of human language.

- **Quasi Character-Level Transformers to Improve Neural Machine Translation on Small Datasets** (Carrión & Casacuberta, 2021): Presents a quasi character-level approach to vocabulary design, aimed at improving NMT in low-resource settings. By reducing vocabulary granularity, the proposed Quasi-Character-based (QCB) model consistently outperforms traditional subword models on small datasets, providing

early evidence for the effectiveness of these vocabularies in low-resource scenarios.

- **Incremental Vocabularies in Machine Translation Through Aligned Embedding Projections** (Carrión & Casacuberta, 2022a): Investigates vocabulary expansion from a zero-shot perspective, focusing on the integration of new embeddings into pre-trained models. The study emphasizes the role of alignment quality and embedding dimensionality, and shows how compositional embeddings, facilitate incremental vocabulary updates, findings that directly support the design of continual vocabularies in this thesis.
- **On the Effectiveness of Quasi Character-Level Models for Machine Translation** (Carrión & Casacuberta, 2022c): Extends this analysis across multiple domains, languages, and neural architectures. It confirms the robustness and generalization capabilities of QCB models while highlighting their computational efficiency compared to purely character-level alternatives, conclusions that reinforce the arguments made in Chapter 5 regarding the vocabulary granularity.

B.1.2 Mitigating Catastrophic Forgetting in NMT

These works focus on continual learning strategies that prevent models from forgetting previously acquired knowledge when new domains or languages are learned.

- **Few-Shot Regularization to Tackle Catastrophic Forgetting in Multilingual Machine Translation** (Carrión & Casacuberta, 2022b): Introduces a hybrid strategy that combines minimal rehearsal with regularization to alleviate forgetting in multilingual NMT. We found that even replaying a small fraction of past data proves to be a highly effective technique for mitigating the forgetting in NMT. The work also proposes a novel loss function integrating few-shot rehearsal, adaptive re-weighting, and knowledge distillation elements that form a central part of the discussion on regularization in Chapter 6.
- **Continual Vocabularies to Tackle the Catastrophic Forgetting Problem in Machine Translation** (Carrión & Casacuberta, 2023b): Explores the intersection between the vocabulary domain and the catastrophic forgetting problem. It introduces a continual vocabulary approach based on compositional embeddings, demonstrating its capacity to maintain cross-domain performance and mitigate the forgetting. This contribu-

tion bridges the discussions in both Chapters 5 and 6, offering a unified perspective on vocabulary evolution in continual learning.

- **Efficient Continual Learning in Neural Machine Translation: A Low-Rank Adaptation Approach** (Carrión & Casacuberta, 2024): Proposes a parameter-efficient approach to task adaptation using Low-Rank Adaptation (LoRA). It introduces a Mixture of LoRA Experts (MoLE) architecture for interactive task-switching strategies, along with a gradient-based regularization technique to integrate new knowledge while minimizing the forgetting in these highly compressed representation space. This contribution is pivotal to the thesis' exploration of efficient continual learning mechanisms.

B.1.3 Tools and Frameworks

These publications describe tools developed or used throughout this work to streamline experimentation and improve reproducibility.

- **The DeepHealth Toolkit: A Unified Framework to Boost Biomedical Applications** (Cancilla et al., 2021): Introduces the DeepHealth Toolkit, built for high-performance computing scenarios in biomedical domains. Although not specific to NMT, its core library (EDDL) served as the basis for AutoNMT's early development, indirectly supporting this thesis' technical foundation.
- **The SELENE Deep Learning Acceleration Framework for Safety-related Applications** (Medina et al., 2022): Details the SELENE framework for deploying AI models in safety-critical systems using RISC-V SoCs. While not related to machine translation, it contributed to understanding efficient model deployment and hardware/software co-design, influencing broader architectural considerations.
- **AutoNMT: A Framework to Streamline the Research of Seq2Seq Models** (Carrión & Casacuberta, 2023a): Describes AutoNMT, a Python framework that streamlines the research of sequence-to-sequence models. This tool was essential for the completion of this thesis.

B.1.4 Other Publications

These contributions originated from interdisciplinary projects carried out during the PhD program, extending beyond the direct scope of this thesis.

- **Automatic Detection of Epileptic Seizures with Recurrent and Convolutional Neural Networks** (Carrión et al., 2022a): Applies deep learning techniques to detect epileptic seizures from EEG data. It validates the DeepHealth Toolkit in a medical context, showcasing the flexibility of the framework later repurposed for NMT experiments.
- **Detection of Pulmonary Conditions Using the DeepHealth Framework** (Carrión et al., 2022b): Explores the detection of COVID-19-related pulmonary conditions using chest X-rays. Though domain-specific, this paper emphasizes the potential of flexible, high-performance deep learning pipelines for image-based classification tasks.

List of Figures

List of Tables

List of Acronyms

AGI Artificial General Intelligence

AI Artificial Intelligence

ASI Artificial Superintelligence

BERTScore Bidirectional Encoder Representations from Transformers Score

BLEU Bilingual Evaluation Understudy

BPE Byte-Pair Encoding

BRNNs Bidirectional Recurrent Neural Networks

BWT Backward Transfer

CE Cross-Entropy

CF Catastrophic Forgetting

ChrF Character F-score

CL Continual Learning

CLM Causal Language Modeling

CLS Complementary Learning Systems

CNNs Convolutional Neural Networks

CoT Chain-of-Thought

DL Deep Learning

ECVL European Computer Vision Library

EDDL European Distributed Deep Learning Library

EWC Elastic Weight Consolidation

FM Forgetting Measure

FT Fine-Tuning

FWT Forward Transfer

GloVe Global Vectors for Word Representation

GRUs Gated Recurrent Units

GUI Graphical User Interface

HPC High-Performance Computing

ICL In-context learning

IL Incremental learning

IMT Interactive Machine Translation

IoU Intersection Over Union

LLMs Large Language Models

LoRA Low-Rank Adaptation

LRMT Low-Resource Machine Translation

LSTMs Long Short-Term Memory networks

LwF Learning without Forgetting

MCP Model Context Protocol

METEOR Metric for Evaluation of Translation with Explicit ORdering

ML Machine Learning

MLM Masked Language Modeling

MNMT Multilingual Neural Machine Translation

MoE Mixture of Experts

MoLE Mixture of LoRA Experts

MQM Multidimensional Quality Metrics

MT Machine Translation

MTL Multi-Task Learning

NLL Negative Log-Likelihood

NLP Natural Language Processing

NMT Neural Machine Translation

OOV Out-of-Vocabulary

PCA Principal Component Analysis

PEFT Parameter-Efficient Fine-Tuning

QCB Quasi-Character-based

QLoRA Quantized Low-Rank Adaptation

RAG Retrieval-Augmented Generation

RL Reinforcement Learning

RNNs Recurrent Neural Networks

SMT Statistical Machine Translation

SOTA State-of-the-Art

TER Translation Edit Rate

ToT Tree-of-Thought

Bibliography

Aharoni, R., Johnson, M., & Firat, O. (2019). Massively multilingual neural machine translation. *CoRR* (cit. on p. 8).

Amodei, D., & Hernandez, D. (2018, May). AI and Compute. (Cit. on p. 166).

Anthropic. (2024a). Introducing the claude 3 model family. (Cit. on p. 164).

Anthropic. (2024b). Model context protocol (mcp): Introduction. (Cit. on p. 165).

Arivazhagan, N., Bapna, A., Firat, O., Lepikhin, D., Johnson, M., Krikun, M., Chen, M. X., Cao, Y., Foster, G. F., Cherry, C., Macherey, W., Chen, Z., & Wu, Y. (2019). Massively multilingual neural machine translation in the wild: Findings and challenges. *CoRR* (cit. on p. 48).

Artetxe, M., Labaka, G., & Agirre, E. (2018, October). Unsupervised statistical machine translation. In E. Riloff, D. Chiang, J. Hockenmaier, & J. Tsujii (Eds.), *Proceedings of the 2018 conference on empirical methods in natural language processing* (pp. 3632–3642). Association for Computational Linguistics. (Cit. on p. 29).

Aschenbrenner, L. (2024, June). Situational awareness: The decade ahead (2024-2034). (Cit. on p. 166).

Ataman, D., & Federico, M. (2018, March). An evaluation of two vocabulary reduction methods for neural machine translation. In C. Cherry & G. Neubig (Eds.), *Proceedings of the 13th conference of the association for machine translation in the Americas (volume 1: Research track)* (pp. 97–110). Association for Machine Translation in the Americas. (Cit. on p. 8).

Bahdanau, D., Brakel, P., Xu, K., Goyal, A., Lowe, R., Pineau, J., Courville, A. C., & Bengio, Y. (2016). An actor-critic algorithm for sequence prediction. *CoRR* (cit. on p. 33).

Bahdanau, D., Cho, K., & Bengio, Y. (2015). Neural machine translation by jointly learning to align and translate. *CoRR* (cit. on pp. 2, 5, 94).

Banar, N., Daelemans, W., & Kestemont, M. (2020). Character-level transformer-based neural machine translation. *CoRR* (cit. on p. 6).

Bapna, A., & Firat, O. (2019, November). Simple, scalable adaptation for neural machine translation. In K. Inui, J. Jiang, V. Ng, & X. Wan (Eds.), *Proceedings of the 2019 conference on empirical methods in natural language processing and the 9th international joint conference on natural language processing (emnlp-ijcnlp)* (pp. 1538–1548). Association for Computational Linguistics. (Cit. on p. 48).

Bar-Hillel, Y. (1953). Some linguistic problems connected with machine translation. *Philosophy of Science*, *20*(3), 217–225 (cit. on p. 1).

Barrault, L., Biesialska, M., Bojar, O., Costa-jussà, M. R., Federmann, C., Graham, Y., Grundkiewicz, R., Haddow, B., Huck, M., Joanis, E., Kocmi, T., Koehn, P., Lo, C.-k., Ljubešić, N., Monz, C., Morishita, M., Nagata, M., Nakazawa, T., Pal, S., … Zampieri, M. (2020). Findings of the 2020 conference on machine translation (WMT20). *Proceedings of the Fifth Conference on Machine Translation*, 1–55 (cit. on pp. 18, 28, 34).

Belinkov, Y., Durrani, N., Dalvi, F., Sajjad, H., & Glass, J. R. (2019). On the linguistic representational power of neural machine translation models. *CoRR* (cit. on p. 64).

Bentivogli, L., Cettolo, M., Federico, M., & Federmann, C. (2018, October). Machine translation human evaluation: An investigation of evaluation based on post-editing and its relation with direct assessment. In M. Turchi,

J. Niehues, & M. Frederico (Eds.), *Proceedings of the 15th international conference on spoken language translation* (pp. 62–69). International Conference on Spoken Language Translation. (Cit. on p. 37).

Bi, Z., Han, K., Liu, C., Tang, Y., & Wang, Y. (2025). Forest-of-thought: Scaling test-time compute for enhancing llm reasoning (cit. on p. 165).

Biesialska, M., Biesialska, K., & Costa-jussà, M. R. (2020). Continual lifelong learning in natural language processing: A survey. *CoRR* (cit. on p. 109).

Bishop, C. M. (2006). *Pattern recognition and machine learning*. Springer. (Cit. on p. 28).

Black, P. E. (2005, February). Dictionary of algorithms and data structures. (Cit. on p. 20).

Blakeman, S., & Mareschal, D. (2019). A complementary learning systems approach to temporal difference learning. (Cit. on p. 5).

Bojanowski, P., Grave, E., Joulin, A., & Mikolov, T. (2016). Enriching word vectors with subword information. *CoRR* (cit. on pp. 99, 103, 104, 113).

Brown, P. F., Pietra, V. J. D., Pietra, S. A. D., & Mercer, R. L. (1993). The mathematics of statistical machine translation: Parameter estimation. *Comput. Linguist.*, *19*(2), 263–311 (cit. on pp. 64, 65).

Brown, T. B., Mann, B., Ryder, N., Subbiah, M., Kaplan, J., Dhariwal, P., Neelakantan, A., Shyam, P., Sastry, G., Askell, A., Agarwal, S., Herbert-Voss, A., Krueger, G., Henighan, T., Child, R., Ramesh, A., Ziegler, D. M., Wu, J., Winter, C., ... Amodei, D. (2020). Language models are few-shot learners. *CoRR* (cit. on pp. 2, 18).

Cancilla, M., Canalini, L., Bolelli, F., Allegretti, S., Carrión, S., Paredes, R., Gómez, J. A., Leo, S., Piras, M. E., Pireddu, L., Badouh, A., Marco-Sola, S., Alvarez, L., Moreto, M., & Grana, C. (2021). The deephealth toolkit: A unified framework to boost biomedical applications. *2020 25th International Conference on Pattern Recognition (ICPR)*, 9881–9888 (cit. on pp. 168, 173).

Cao, Y., Wei, H.-R., Chen, B., & Wan, X. (2021, June). Continual learning for neural machine translation. In K. Toutanova, A. Rumshisky, L. Zettlemoyer, D. Hakkani-Tur, I. Beltagy, S. Bethard, R. Cotterell, T. Chakraborty, & Y. Zhou (Eds.), *Proceedings of the 2021 conference of the north american chapter of the association for computational linguistics: Human language technologies* (pp. 3964–3974). Association for Computational Linguistics. (Cit. on pp. 7, 49, 62, 109).

Carpenter, G. A., & Grossberg, S. (1987). A massively parallel architecture for a self-organizing neural pattern recognition machine. *Computer Vision, Graphics, and Image Processing*, *37*(1), 54–115 (cit. on pp. 5, 43).

Carrión, S., & Casacuberta, F. (2021). Quasi character-level transformers to improve neural machine translation on small datasets. *2021 Eighth International Conference on Social Network Analysis, Management and Security (SNAMS)*, 1–6 (cit. on pp. 9, 77, 78, 159, 171).

Carrión, S., & Casacuberta, F. (2022a). Incremental vocabularies in machine translation through aligned embedding projections. In A. J. Pinho, P. Georgieva, L. F. Teixeira, & J. A. Sánchez (Eds.), *Pattern recognition and image analysis* (pp. 27–40). Springer International Publishing. (Cit. on pp. 9, 99, 160, 172).

Carrión, S., & Casacuberta, F. (2022b, September). Few-shot regularization to tackle catastrophic forgetting in multilingual machine translation. In K. Duh & F. Guzmán (Eds.), *Proceedings of the 15th biennial conference of the association for machine translation in the americas (volume 1: Research track)* (pp. 188–199). Association for Machine Translation in the Americas. (Cit. on pp. 9, 48, 127, 130, 132, 161, 172).

Carrión, S., & Casacuberta, F. (2022c, September). On the effectiveness of quasi character-level models for machine translation. In K. Duh & F. Guzmán (Eds.), *Proceedings of the 15th biennial conference of the association for machine translation in the americas (volume 1: Research track)* (pp. 131–143). Association for Machine Translation in the Americas. (Cit. on pp. 9, 77, 78, 159, 172).

Carrión, S., & Casacuberta, F. (2023a). Autonmt: A framework to streamline the research of seq2seq models. (Cit. on pp. 53, 57, 167, 173).

Carrión, S., & Casacuberta, F. (2023b). Continual vocabularies to tackle the catastrophic forgetting problem in machine translation. In A. Pertusa, A. J. Gallego, J. A. Sánchez, & I. Domingues (Eds.), *Pattern recognition and image analysis* (pp. 94–107). Springer Nature Switzerland. (Cit. on pp. 9, 109, 113, 122, 124, 160, 172).

Carrión, S., & Casacuberta, F. (2024). Efficient continual learning in neural machine translation: A low-rank adaptation approach. *Preprint* (cit. on pp. 10, 48, 136, 138, 140, 141, 145, 151, 153, 161, 173).

Carrión, S., López-Chilet, Á., Martínez-Bernia, J., Coll-Alonso, J., Chorro-Juan, D., & Gómez, J. A. (2022a). Automatic detection of epileptic seizures with recurrent and convolutional neural networks. In P. L. Mazzeo, E. Frontoni, S. Sclaroff, & C. Distante (Eds.), *Image analysis and processing. iciap 2022 workshops* (pp. 522–532). Springer International Publishing. (Cit. on p. 174).

Carrión, S., López-Chilet, Á., Martínez-Bernia, J., Coll-Alonso, J., Chorro-Juan, D., & Gómez, J. A. (2022b). Detection of pulmonary conditions using the deephealth framework. In P. L. Mazzeo, E. Frontoni, S. Sclaroff, & C. Distante (Eds.), *Image analysis and processing. iciap 2022 workshops* (pp. 557–566). Springer International Publishing. (Cit. on p. 174).

Cettolo, M., Girardi, C., & Federico, M. (2012). WIT3: Web inventory of transcribed and translated talks. *Proceedings of the 16th Annual conference of the EAMT*, 261–268 (cit. on p. 52).

Chen, Z., Liu, B., Brachman, R., Stone, P., & Rossi, F. (2018). *Lifelong machine learning* (2nd). Morgan & Claypool Publishers. (Cit. on pp. 30, 41, 44).

Cheng, Y., Xu, W., He, Z., He, W., Wu, H., Sun, M., & Liu, Y. (2016, August). Semi-supervised learning for neural machine translation. In K. Erk & N. A. Smith (Eds.), *Proceedings of the 54th annual meeting of the association for computational linguistics (volume 1: Long papers)* (pp. 1965–1974). Association for Computational Linguistics. (Cit. on p. 29).

Cherry, C., Foster, G. F., Bapna, A., Firat, O., & Macherey, W. (2018). Revisiting character-based neural machine translation with capacity and compression. *CoRR* (cit. on pp. 7, 39, 111).

Chiang, W.-L., Zheng, L., Sheng, Y., Angelopoulos, A. N., Li, T., Li, D., Zhang, H., Zhu, B., Jordan, M., Gonzalez, J. E., & Stoica, I. (2024). Chatbot arena: An open platform for evaluating llms by human preference. (Cit. on pp. 164, 165).

Cho, K., van Merriënboer, B., Gulcehre, C., Bahdanau, D., Bougares, F., Schwenk, H., & Bengio, Y. (2014). Learning phrase representations using RNN encoder–decoder for statistical machine translation. *Proceedings of the 2014 Conference on EMNLP*, 1724–1734 (cit. on pp. 55, 94).

Chollet, F. (2019). On the measure of intelligence. (Cit. on p. 165).

Chomsky, N. (1956). Three models for the description of language. *IRE Transactions on Information Theory*, *2*(3), 113–124 (cit. on pp. 64, 65).

Chowdhery, A., Narang, S., Devlin, J., Bosma, M., Mishra, G., Roberts, A., Barham, P., Chung, H. W., Sutton, C., Gehrmann, S., Schuh, P., Shi, K., Tsvyashchenko, S., Maynez, J., Rao, A., Barnes, P., Tay, Y., Shazeer, N., Prabhakaran, V., … Fiedel, N. (2022). Palm: Scaling language modeling with pathways. (Cit. on p. 18).

Clark, J. H., Garrette, D., Turc, I., & Wieting, J. (2021). CANINE: pre-training an efficient tokenization-free encoder for language representation. *CoRR* (cit. on pp. 6, 26).

Committee, A. L. P. A. (1966). *Languages and machines: Computers in translation and linguistics (alpac report)*. National Academy of Sciences, National Research Council. (Cit. on p. 1).

Conneau, A., Khandelwal, K., Goyal, N., Chaudhary, V., Wenzek, G., Guzmán, F., Grave, E., Ott, M., Zettlemoyer, L., & Stoyanov, V. (2019). Unsupervised cross-lingual representation learning at scale. *CoRR* (cit. on pp. 8, 48).

Conneau, A., Schwenk, H., Barrault, L., & LeCun, Y. (2016). Very deep convolutional networks for natural language processing. *CoRR* (cit. on p. 55).

Contributors, T. (2013). Tatoeba: Collection of sentences and translations [Accessed: 2024-12-13]. (Cit. on p. 52).

Costa-jussà, M. R., Escolano, C., & Fonollosa, J. A. R. (2017, September). Byte-based neural machine translation. In M. Faruqui, H. Schuetze, I. Trancoso, & Y. Yaghoobzadeh (Eds.), *Proceedings of the first workshop on subword and character level models in NLP* (pp. 154–158). Association for Computational Linguistics. (Cit. on pp. 26, 67, 77).

Costa-jussà, M. R., & Fonollosa, J. A. R. (2016). Character-based neural machine translation. *Proceedings of the 54th Annual Meeting of the ACL (Volume 2: Short Papers)*, 357–361 (cit. on p. 6).

De Lange, M., Aljundi, R., Masana, M., Parisot, S., Jia, X., Leonardis, A., Slabaugh, G., & Tuytelaars, T. (2022). A continual learning survey: Defying forgetting in classification tasks. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, *44*(7), 3366–3385 (cit. on p. 44).

Deepl translator [Accessed: 2023-11-15]. (2023). (Cit. on pp. 52, 140).

Dettmers, T., Pagnoni, A., Holtzman, A., & Zettlemoyer, L. (2023). Qlora: Efficient finetuning of quantized llms. (Cit. on p. 145).

Devlin, J., Chang, M., Lee, K., & Toutanova, K. (2018). BERT: pre-training of deep bidirectional transformers for language understanding. *CoRR* (cit. on pp. 36, 99).

Dickson, B. (2024, December). OpenAI's o3 shows remarkable progress on ARC-AGI, sparking debate on AI reasoning. (Cit. on p. 165).

Dong, D., Wu, H., He, W., Yu, D., & Wang, H. (2015, July). Multi-task learning for multiple language translation. In C. Zong & M. Strube (Eds.), *Proceedings of the 53rd annual meeting of the association for computational linguistics and the 7th international joint conference on natural language processing (volume 1: Long papers)* (pp. 1723–1732). Association for Computational Linguistics. (Cit. on p. 31).

Dostert, L., Garvin, P., C., H., & P., S. (1954). The georgetown-ibm experiment (cit. on p. 1).

Elliott, D., Frank, S., Sima'an, K., & Specia, L. (2016). Multi30K: Multilingual English-German image descriptions. *Proceedings of the 5th Workshop on Vision and Language*, 70–74 (cit. on pp. 52, 100).

Fernando, A., & Ranathunga, S. (2022). Data augmentation to address out-of-vocabulary problem in low-resource sinhala-english neural machine translation. (Cit. on p. 62).

Franchi, G., Bursuc, A., Aldea, E., Dubuisson, S., & Bloch, I. (2019). TRADI: tracking deep neural network weight distributions. *CoRR* (cit. on p. 149).

Garcia, X., Constant, N., Parikh, A. P., & Firat, O. (2021). Towards continual learning for multilingual machine translation via vocabulary substitution (cit. on pp. 49, 109).

Gehring, J., Auli, M., Grangier, D., Yarats, D., & Dauphin, Y. N. (2017). Convolutional sequence to sequence learning. *CoRR* (cit. on pp. 16, 94).

Gillick, D., Brunk, C., Vinyals, O., & Subramanya, A. (2015). Multilingual language processing from bytes. *CoRR* (cit. on p. 26).

Google DeepMind. (2025). Gemini 2.5 pro: Our most intelligent ai model. (Cit. on p. 164).

Gowda, T., & May, J. (2020). Finding the optimal vocabulary size for neural machine translation. *Findings of the ACL: EMNLP 2020*, 3955–3964 (cit. on pp. 26, 64, 75, 112).

Graham, Y., Baldwin, T., Moffat, A., & Zobel, J. (2013, August). Continuous measurement scales in human evaluation of machine translation. In A. Pareja-Lora, M. Liakata, & S. Dipper (Eds.), *Proceedings of the 7th linguistic annotation workshop and interoperability with discourse* (pp. 33–41). Association for Computational Linguistics. (Cit. on p. 37).

Han, Y., Huang, G., Song, S., Yang, L., Wang, H., & Wang, Y. (2021). Dynamic neural networks: A survey. *CoRR* (cit. on p. 8).

Haque, R., Liu, C.-H., & Way, A. (2021). Recent advances of low-resource neural machine translation. *Machine Translation*, *35*, 451–474 (cit. on p. 39).

He, D., Xia, Y., Qin, T., Wang, L., Yu, N., Liu, T.-Y., & Ma, W.-Y. (2016). Dual learning for machine translation. In D. Lee, M. Sugiyama, U. Luxburg, I. Guyon, & R. Garnett (Eds.), *Advances in neural information processing systems* (Vol. 29). Curran Associates, Inc. (Cit. on p. 29).

Hebb, D. O. (1949). The organization of behavior: A neuropsychological theory (cit. on p. 118).

Hendrycks, D., Burns, C., Kadavath, S., Arora, A., Basart, S., Tang, E., Song, D., & Steinhardt, J. (2021). Measuring mathematical problem solving with the math dataset. (Cit. on p. 165).

Hinder, F., Vaquet, V., & Hammer, B. (2023). A remark on concept drift for dependent data. (Cit. on p. 43).

Hinton, G., Vinyals, O., & Dean, J. (2015). Distilling the knowledge in a neural network. (Cit. on pp. 6, 133).

Hochreiter, S., & Schmidhuber, J. (1997). Long short-term memory. *Neural computation*, *9*, 1735–80 (cit. on pp. 2, 15, 55, 64, 70).

Holtzman, A., Buys, J., Forbes, M., & Choi, Y. (2019). The curious case of neural text degeneration. *CoRR* (cit. on p. 22).

Houlsby, N., Giurgiu, A., Jastrzebski, S., Morrone, B., de Laroussilhe, Q., Gesmundo, A., Attariyan, M., & Gelly, S. (2019). Parameter-efficient transfer learning for NLP. *CoRR* (cit. on pp. 7, 32, 136).

Hu, E. J., Shen, Y., Wallis, P., Allen-Zhu, Z., Li, Y., Wang, S., & Chen, W. (2021). Lora: Low-rank adaptation of large language models. *CoRR* (cit. on pp. 7, 8, 30, 32, 136).

Jacobs, R. A., Jordan, M. I., Nowlan, S. J., & Hinton, G. E. (1991). Adaptive mixtures of local experts. *Neural Computation*, *3*(1), 79–87 (cit. on p. 138).

Jean, S., Cho, K., Memisevic, R., & Bengio, Y. (2014). On using very large target vocabulary for neural machine translation. *CoRR* (cit. on p. 27).

Johnson, M., Schuster, M., Le, Q. V., Krikun, M., Wu, Y., Chen, Z., Thorat, N., Viégas, F. B., Wattenberg, M., Corrado, G., Hughes, M., & Dean, J. (2016). Google's multilingual neural machine translation system: Enabling zero-shot translation. *CoRR* (cit. on p. 98).

Jones, N. (2025). How should we test ai for human-level intelligence? openai's o3 electrifies quest. *Nature*, *637*(8047), 774–775. https://doi.org/10.1038/d41586-025-00110-6 (cit. on p. 165).

Ke, Z., & Liu, B. (2023). Continual learning of natural language processing tasks: A survey. (Cit. on p. 109).

Kemker, R., & Kanan, C. (2017). Fearnet: Brain-inspired model for incremental learning. *CoRR* (cit. on p. 47).

Kim, Y., Jernite, Y., Sontag, D. A., & Rush, A. M. (2015). Character-aware neural language models. *CoRR* (cit. on p. 25).

Kingma, D. P., & Ba, J. (2015). Adam: A method for stochastic optimization. In Y. Bengio & Y. LeCun (Eds.), *3rd international conference on learning representations, ICLR 2015, san diego, ca, usa, may 7-9, 2015, conference track proceedings*. (Cit. on p. 56).

Kirkpatrick, J., Pascanu, R., Rabinowitz, N. C., Veness, J., Desjardins, G., Rusu, A. A., Milan, K., Quan, J., Ramalho, T., Grabska-Barwinska, A., Hassabis, D., Clopath, C., Kumaran, D., & Hadsell, R. (2016). Overcoming catastrophic forgetting in neural networks. *CoRR* (cit. on pp. 6, 8, 29, 30, 46, 48, 119).

Kitaev, N., Kaiser, Ł., & Levskaya, A. (2020). Reformer: The efficient transformer (cit. on p. 84).

Klein, G., Kim, Y., Deng, Y., Senellart, J., & Rush, A. (2017). OpenNMT: Open-source toolkit for neural machine translation. *Proceedings of ACL 2017, System Demonstrations*, 67–72 (cit. on p. 167).

Kocmi, T., Avramidis, E., Bawden, R., Bojar, O., Dvorkovich, A., Federmann, C., Fishel, M., Freitag, M., Gowda, T., Grundkiewicz, R., Haddow, B., Koehn, P., Marie, B., Monz, C., Morishita, M., Murray, K., Nagata, M., Nakazawa, T., Popel, M., ... Shmatova, M. (2023, December). Findings of the 2023 conference on machine translation (WMT23): LLMs are here but not quite there yet. In P. Koehn, B. Haddow, T. Kocmi, & C. Monz (Eds.), *Proceedings of the eighth conference on machine translation* (pp. 1–42). Association for Computational Linguistics. (Cit. on pp. 28, 34, 55).

Koehn, P. (2005). Europarl: A Parallel Corpus for Statistical Machine Translation. *Conference Proceedings: the tenth Machine Translation Summit*, 79–86 (cit. on p. 52).

Koehn, P. (2009). *Statistical machine translation*. Cambridge University Press. (Cit. on p. 2).

Koehn, P., Hoang, H., Birch, A., Callison-Burch, C., Federico, M., Bertoldi, N., Cowan, B., Shen, W., Moran, C., Zens, R., Dyer, C., Bojar, O., Constantin, A., & Herbst, E. (2007). Moses: Open source toolkit for statistical machine translation. *Proceedings of the 45th Annual Meeting of the ACL on Interactive Poster and Demonstration Sessions*, 177–180 (cit. on pp. 53, 54).

Koehn, P., & Knowles, R. (2017, August). Six challenges for neural machine translation. In T. Luong, A. Birch, G. Neubig, & A. Finch (Eds.), *Proceedings of the first workshop on neural machine translation* (pp. 28–39). Association for Computational Linguistics. (Cit. on pp. 23, 27, 62).

Kudo, T. (2018). Subword regularization: Improving neural network translation models with multiple subword candidates. *Proceedings of the 56th Annual Meeting of the ACL (Volume 1: Long Papers)*, 66–75 (cit. on pp. 6, 24, 27, 90).

Kudo, T., & Richardson, J. (2018). Sentencepiece: A simple and language independent subword tokenizer and detokenizer for neural text processing. In E. Blanco & W. Lu (Eds.), *Proceedings of the 2018 conference on empirical methods in natural language processing, EMNLP 2018: System demonstrations, brussels, belgium, october 31 - november 4, 2018* (pp. 66–71). (Cit. on pp. 6, 24, 54).

Kurzweil, R. (2024). *The singularity is nearer*. Penguin Random House. (Cit. on p. 166).

Lample, G., Ott, M., Conneau, A., Denoyer, L., & Ranzato, M. (2018, October). Phrase-based & neural unsupervised machine translation. In E. Riloff, D. Chiang, J. Hockenmaier, & J. Tsujii (Eds.), *Proceedings of the 2018 conference on empirical methods in natural language processing* (pp. 5039–5049). Association for Computational Linguistics. (Cit. on p. 29).

Lange, M. D., van de Ven, G., & Tuytelaars, T. (2023). Continual evaluation for lifelong learning: Identifying the stability gap. (Cit. on p. 49).

Lavie, A., & Agarwal, A. (2007). Meteor: An automatic metric for mt evaluation with high levels of correlation with human judgments. *Proceedings of the Second Workshop on Statistical Machine Translation*, 228–231 (cit. on p. 36).

LeCun, Y. (2022). A path towards autonomous machine intelligence [Version 0.9.2, June 2022, available at openreview.net]. (Cit. on p. 166).

Lee, J., Cho, K., & Hofmann, T. (2016). Fully character-level neural machine translation without explicit segmentation. *CoRR* (cit. on pp. 6, 25).

Lee, J., Yoon, J., Yang, E., & Hwang, S. J. (2017). Lifelong learning with dynamically expandable networks. *CoRR* (cit. on pp. 7, 8).

Lewis, M. (2023). Compositional vector semantics in spiking neural networks. In *Trends and challenges in cognitive modeling* (pp. 131–146). Springer International Publishing. (Cit. on p. 99).

Lewis, P., Perez, E., Piktus, A., Petroni, F., Karpukhin, V., Goyal, N., Küttler, H., Lewis, M., Yih, W.-t., Rocktäschel, T., Riedel, S., & Kiela, D. (2020). Retrieval-augmented generation for knowledge-intensive nlp tasks. *Proceedings of the 34th International Conference on Neural Information Processing Systems* (cit. on p. 165).

Li, Z., & Hoiem, D. (2016). Learning without forgetting. *CoRR* (cit. on pp. 7, 8, 30, 46).

Liang, Y., Meng, F., Wang, J., Xu, J., Chen, Y., & Zhou, J. (2024, August). Continual learning with semi-supervised contrastive distillation for incremental neural machine translation. In L.-W. Ku, A. Martins, & V. Srikumar (Eds.), *Proceedings of the 62nd annual meeting of the association for computational linguistics (volume 1: Long papers)* (pp. 10914–10928). Association for Computational Linguistics. (Cit. on p. 30).

Littlestone, N., & Warmuth, M. (1994). The weighted majority algorithm. *Information and Computation*, *108*(2), 212–261 (cit. on p. 29).

Liu, Z., Winata, G. I., & Fung, P. (2021). Continual mixed-language pre-training for extremely low-resource neural machine translation. (Cit. on p. 48).

Lommel, A., Burchardt, A., & Uszkoreit, H. (2014). Multidimensional quality metrics (mqm): A framework for declaring and describing translation quality metrics. *Tradumàtica: tecnologies de la traducció*, *0*, 455–463 (cit. on p. 38).

Lopez-Paz, D., & Ranzato, M. (2017). Gradient episodic memory for continuum learning. *CoRR* (cit. on p. 7).

Loshchilov, I., & Hutter, F. (2017). Fixing weight decay regularization in adam. *CoRR* (cit. on p. 56).

Luong, M.-T., Le, Q. V., Sutskever, I., Vinyals, O., & Kaiser, L. (2015). Multi-task sequence to sequence learning. *CoRR* (cit. on p. 31).

Luong, T., & Manning, C. D. (2016). Achieving open vocabulary neural machine translation with hybrid word-character models. *Proceedings of the 54th Annual Meeting of the ACL (Volume 1: Long Papers)*, 1054–1063 (cit. on p. 27).

Luong, T., Pham, H., & Manning, C. D. (2015). Effective approaches to attention-based neural machine translation. *CoRR* (cit. on p. 27).

Luong, T., Sutskever, I., Le, Q. V., Vinyals, O., & Zaremba, W. (2014). Addressing the rare word problem in neural machine translation. *CoRR* (cit. on pp. 23, 27, 62).

Mcclelland, J., Mcnaughton, B., & O'Reilly, R. (1995). Why there are complementary learning systems in the hippocampus and neocortex: Insights from the successes and failures of connectionist models of learning and memory. *Psychological review*, *102*, 419–57 (cit. on pp. 5, 29, 47, 117).

McCloskey, M., & Cohen, N. J. (1989). Catastrophic interference in connectionist networks: The sequential learning problem. In G. H. Bower (Ed.). Academic Press. (Cit. on pp. 5, 43).

Medina, L., Carrión, S., Andreu, P., Picornell, T., Flich, J., Hernàndez, C., Sandoval, M., Sainz, M., Lefebvre, C.-A., Rönnbäck, M., Matschnig, M., Wess, M., & Taucher, H. (2022). The selene deep learning acceleration framework for safety-related applications. *2022 Design, Automation & Test in Europe Conference & Exhibition (DATE)*, 636–639 (cit. on pp. 169, 173).

Meta AI. (2025). Llama 4: Multimodal intelligence. (Cit. on p. 164).

Mielke, S. J., Alyafeai, Z., Salesky, E., Raffel, C., Dey, M., Gallé, M., Raja, A., Si, C., Lee, W. Y., Sagot, B., & Tan, S. (2021). Between words and characters: A brief history of open-vocabulary modeling and tokenization in NLP. *CoRR* (cit. on p. 75).

Moi, A., & Patry, N. (2023, April). *HuggingFace's Tokenizers* (Version 0.13.4). (Cit. on p. 53).

Moosa, I. M., Zhang, R., & Yin, W. (2024). Mt-ranker: Reference-free machine translation evaluation by inter-system ranking. (Cit. on p. 37).

Mullov, C., Pham, Q., & Waibel, A. (2024, August). Decoupled vocabulary learning enables zero-shot translation from unseen languages. In L.-W. Ku, A. Martins, & V. Srikumar (Eds.), *Proceedings of the 62nd annual meeting of the association for computational linguistics (volume 1: Long papers)* (pp. 6693–6709). Association for Computational Linguistics. (Cit. on p. 98).

Neubig, G., & Hu, J. (2018, October). Rapid adaptation of neural machine translation to new languages. In E. Riloff, D. Chiang, J. Hockenmaier, & J. Tsujii (Eds.), *Proceedings of the 2018 conference on empirical methods in natural language processing* (pp. 875–880). Association for Computational Linguistics. (Cit. on p. 31).

Neubig, G., Watanabe, T., Mori, S., & Kawahara, T. (2013). Substring-based machine translation. *Machine Translation*, *27*(2), 139–166 (cit. on p. 64).

Neves, M., Yepes, A. J., & Névéol, A. (2016). The scielo corpus: A parallel corpus of scientific publications for biomedicine. *Proceedings of the Tenth International Conference on Language Resources and Evaluation (LREC'16)*, 2942–2948 (cit. on p. 52).

Nguyen, K., III, H. D., & Boyd-Graber, J. L. (2017). Reinforcement learning for bandit neural machine translation with simulated human feedback. *CoRR* (cit. on p. 32).

OpenAI, Achiam, J., Adler, S., Agarwal, S., Ahmad, L., Akkaya, I., Aleman, F. L., Almeida, D., Altenschmidt, J., Altman, S., Anadkat, S., Avila, R., Babuschkin, I., Balaji, S., Balcom, V., Baltescu, P., Bao, H., Bavarian, M., Belgum, J., ... Zoph, B. (2024). Gpt-4 technical report. (Cit. on pp. 2, 48, 164).

Oquab, M., Bottou, L., Laptev, I., & Sivic, J. (2014). Learning and transferring mid-level image representations using convolutional neural networks. *2014 IEEE Conference on Computer Vision and Pattern Recognition*, 1717–1724 (cit. on p. 31).

Ott, M., Edunov, S., Baevski, A., Fan, A., Gross, S., Ng, N., Grangier, D., & Auli, M. (2019). Fairseq: A fast, extensible toolkit for sequence modeling. *CoRR* (cit. on pp. 57, 167).

Papineni, K., Roukos, S., Ward, T., & Zhu, W.-J. (2002). Bleu: A method for automatic evaluation of machine translation. *Proceedings of the 40th Annual Meeting on ACL*, 311–318 (cit. on pp. 33, 59).

Parisi, G. I., Kemker, R., Part, J. L., Kanan, C., & Wermter, S. (2019). Continual lifelong learning with neural networks: A review. *Neural Networks*, *113*, 54–71 (cit. on pp. 30, 41).

Parisi, G. I., Kemker, R., Part, J. L., Kanan, C., & Wermter, S. (2018). Continual lifelong learning with neural networks: A review. *CoRR* (cit. on pp. 29, 30).

Pennington, J., Socher, R., & Manning, C. (2014). GloVe: Global vectors for word representation, 1532–1543 (cit. on p. 100).

Peris, Á., & Casacuberta, F. (2019). Online learning for effort reduction in interactive neural machine translation. (Cit. on pp. 29, 38).

Peters, M. E., Neumann, M., Iyyer, M., Gardner, M., Clark, C., Lee, K., & Zettlemoyer, L. (2018). Deep contextualized word representations. *CoRR* (cit. on p. 99).

Pfeiffer, J., Kamath, A., Rücklé, A., Cho, K., & Gurevych, I. (2021, April). AdapterFusion: Non-destructive task composition for transfer learning. In P. Merlo, J. Tiedemann, & R. Tsarfaty (Eds.), *Proceedings of the 16th conference of the european chapter of the association for computational linguistics: Main volume* (pp. 487–503). Association for Computational Linguistics. (Cit. on pp. 8, 30, 32).

Pham, Q., Liu, C., & Hoi, S. C. H. (2021). Dualnet: Continual learning, fast and slow. *CoRR* (cit. on p. 5).

Popović, M. (2015). ChrF: Character n-gram F-score for automatic MT evaluation. *Proceedings of the Tenth WMT*, 392–395 (cit. on pp. 34, 59).

Popović, M. (2018). Error classification and analysis for machine translation quality assessment. In J. Moorkens, S. Castilho, F. Gaspari, & S. Doherty (Eds.), *Translation quality assessment: From principles to practice* (pp. 129–158). Springer International Publishing. (Cit. on p. 37).

Post, M. (2018). A call for clarity in reporting BLEU scores. *Proceedings of the Third Conference on Machine Translation: Research Papers*, 186–191 (cit. on p. 59).

Pratt, L. Y. (1992). Discriminability-based transfer between neural networks. In S. Hanson, J. Cowan, & C. Giles (Eds.), *Advances in neural information processing systems* (Vol. 5). Morgan-Kaufmann. (Cit. on p. 31).

Qu, H., Rahmani, H., Xu, L., Williams, B., & Liu, J. (2024). Recent advances of continual learning in computer vision: An overview. (Cit. on p. 45).

Radford, A., Wu, J., Child, R., Luan, D., Amodei, D., & Sutskever, I. (2019). Language models are unsupervised multitask learners (cit. on pp. 18, 22).

Ranzato, M., Chopra, S., Auli, M., & Zaremba, W. (2016). Sequence level training with recurrent neural networks. (Cit. on p. 32).

Raunak, V., Dalmia, S., Gupta, V., & Metze, F. (2020). On long-tailed phenomena in neural machine translation. *CoRR* (cit. on pp. 65, 70).

Rein, D., Hou, B. L., Stickland, A. C., Petty, J., Pang, R. Y., Dirani, J., Michael, J., & Bowman, S. R. (2023). Gpqa: A graduate-level google-proof q&a benchmark. (Cit. on p. 165).

Robbins, H., & Monro, S. (1951). A Stochastic Approximation Method. *The Annals of Mathematical Statistics*, *22*(3), 400–407 (cit. on p. 56).

Robins, A. (1995). Catastrophic forgetting, rehearsal and pseudorehearsal. *Connection Science*, *7*(2), 123–146 (cit. on pp. 45, 48).

Rodríguez, N. D., Lomonaco, V., Filliat, D., & Maltoni, D. (2018). Don't forget, there is more than forgetting: New metrics for continual learning. *CoRR* (cit. on p. 49).

Rumelhart, D. E., Hinton, G. E., & Williams, R. J. (1986). Learning internal representations by error propagation. In *Parallel distributed processing: Explorations in the microstructure of cognition, vol. 1: Foundations* (pp. 318–362). MIT Press. (Cit. on pp. 2, 14, 55).

Rusu, A. A., Rabinowitz, N. C., Desjardins, G., Soyer, H., Kirkpatrick, J., Kavukcuoglu, K., Pascanu, R., & Hadsell, R. (2016). Progressive neural networks. *CoRR* (cit. on pp. 7, 47).

Sahin, C. S., Caceres, R. S., Oselio, B., & Campbell, W. M. (2017). Consistent alignment of word embedding models. (Cit. on p. 99).

Sato, S., Sakuma, J., Yoshinaga, N., Toyoda, M., & Kitsuregawa, M. (2020, November). Vocabulary adaptation for domain adaptation in neural machine translation. In T. Cohn, Y. He, & Y. Liu (Eds.), *Findings of the*

association for computational linguistics: Emnlp 2020 (pp. 4269–4279). Association for Computational Linguistics. (Cit. on pp. 7, 62).

Schwarz, J., Luketina, J., Czarnecki, W. M., Grabska-Barwinska, A., Teh, Y. W., Pascanu, R., & Hadsell, R. (2018). Progress & compress: A scalable framework for continual learning. (Cit. on p. 30).

Schwenk, H., Chaudhary, V., Sun, S., Gong, H., & Guzmán, F. (2019). Wikimatrix: Mining 135m parallel sentences in 1620 language pairs from wikipedia. *CoRR* (cit. on p. 52).

Sennrich, R., Haddow, B., & Birch, A. (2015). Improving neural machine translation models with monolingual data. *CoRR* (cit. on pp. 29, 39).

Sennrich, R., Haddow, B., & Birch, A. (2016, August). Neural machine translation of rare words with subword units. In K. Erk & N. A. Smith (Eds.), *Proceedings of the 54th annual meeting of the association for computational linguistics (volume 1: Long papers)* (pp. 1715–1725). Association for Computational Linguistics. (Cit. on pp. 6, 23, 24, 27, 62, 64).

Sennrich, R., & Zhang, B. (2019). Revisiting low-resource neural machine translation: A case study. *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, 211–221 (cit. on pp. 7, 112).

Shannon, C. E. (1948). A mathematical theory of communication. *The Bell System Technical Journal*, *27*(3), 379–423 (cit. on pp. 64, 67).

Shen, Y., Song, K., Tan, X., Li, D., Lu, W., & Zhuang, Y. (2023). Hugginggpt: Solving ai tasks with chatgpt and its friends in hugging face. (Cit. on p. 165).

Shin, H., Lee, J. K., Kim, J., & Kim, J. (2017). Continual learning with deep generative replay. *CoRR* (cit. on pp. 7, 8, 30, 45, 48).

Smith, J. R., Saint-Amand, H., Plamada, M., Koehn, P., Callison-Burch, C., & Lopez, A. (2013). Dirt cheap web-scale parallel text from the Common Crawl. *Proceedings of the 51st Annual Meeting of the ACL (Volume 1: Long Papers)*, 1374–1383 (cit. on p. 52).

Snover, M., Dorr, B., Schwartz, R., Micciulla, L., & Makhoul, J. (2006). A study of translation edit rate with targeted human annotation. *Proceedings of the 7th Conference of the AMTA: Technical Papers*, 223–231 (cit. on pp. 35, 59).

Soutif–Cormerais, A., Carta, A., Cossu, A., Hurtado, J., Hemati, H., Lomonaco, V., & de Weijer, J. V. (2023). A comprehensive empirical evaluation on online continual learning. (Cit. on p. 49).

Steinberger, R., Pouliquen, B., Widiger, A., Ignat, C., Erjavec, T., Tufis, D., & Varga, D. (2006). The jrc-acquis: A multilingual aligned parallel corpus with 20+ languages. *CoRR* (cit. on p. 52).

Sundermeyer, M., Alkhouli, T., Wuebker, J., & Ney, H. (2014). Translation modeling with bidirectional recurrent neural networks. *Proceedings of the 2014 Conference on EMNLP*, 14–25 (cit. on pp. 15, 55).

Sutskever, I., Vinyals, O., & Le, Q. V. (2014). Sequence to sequence learning with neural networks. In Z. Ghahramani, M. Welling, C. Cortes, N. Lawrence, & K. Q. Weinberger (Eds.), *Nips* (Vol. 27). (Cit. on pp. 2, 5, 64, 94).

Tay, Y., Dehghani, M., Abnar, S., Shen, Y., Bahri, D., Pham, P., Rao, J., Yang, L., Ruder, S., & Metzler, D. (2020). Long range arena: A benchmark for efficient transformers. *CoRR* (cit. on p. 18).

Technologies, H. P. L. (2020). Sacremoses: A python port of moses' tokenization scripts [Version 0.1.0]. (Cit. on p. 53).

Tiedemann, J. (2012, May). Parallel data, tools and interfaces in OPUS. In N. Calzolari, K. Choukri, T. Declerck, M. U. Doğan, B. Maegaard, J. Mariani, A. Moreno, J. Odijk, & S. Piperidis (Eds.), *Proceedings of the eighth international conference on language resources and evaluation (LREC'12)* (pp. 2214–2218). European Language Resources Association (ELRA). (Cit. on p. 52).

Toma, P. (1970). Systran as a machine translation system (cit. on p. 1).

van de Ven, G. M., Soures, N., & Kudithipudi, D. (2024). Continual learning and catastrophic forgetting. (Cit. on p. 119).

van der Maaten, L., & Hinton, G. (2008). Visualizing data using t-SNE. *Journal of Machine Learning Research*, *9*, 2579–2605 (cit. on p. 102).

Vaswani, A., Shazeer, N., Parmar, N., Uszkoreit, J., Jones, L., Gomez, A. N., Kaiser, L., & Polosukhin, I. (2017). Attention is all you need. (Cit. on pp. 2, 5, 16, 55, 64, 94).

Vilar, D., Peter, J.-T., & Ney, H. (2007). Can we translate letters? *Proceedings of the Second WMT*, 33–39 (cit. on pp. 64, 67, 77).

Wang, C., Cho, K., & Gu, J. (2019). Neural machine translation with byte-level subwords. *CoRR* (cit. on p. 26).

Wang, S., Li, B. Z., Khabsa, M., Fang, H., & Ma, H. (2020). Linformer: Self-attention with linear complexity. *CoRR* (cit. on p. 84).

Wang, Y., Ma, X., Zhang, G., Ni, Y., Chandra, A., Guo, S., Ren, W., Arulraj, A., He, X., Jiang, Z., Li, T., Ku, M., Wang, K., Zhuang, A., Fan, R., Yue, X., & Chen, W. (2024). Mmlu-pro: A more robust and challenging multi-task language understanding benchmark. (Cit. on p. 165).

Williams, R. J. (1992). Simple statistical gradient-following algorithms for connectionist reinforcement learning. *Machine Learning*, *8*(3), 229–256 (cit. on p. 33).

Winata, G. I., Xie, L., Radhakrishnan, K., Wu, S., Jin, X., Cheng, P., Kulkarni, M., & Preotiuc-Pietro, D. (2023). Overcoming catastrophic forgetting in massively multilingual continual learning. (Cit. on p. 48).

Wolpert, D., & Macready, W. (1997). No free lunch theorems for optimization. *IEEE Transactions on Evolutionary Computation*, *1*(1), 67–82 (cit. on p. 65).

Wu, J., Liu, Y., & Zong, C. (2024). F-malloc: Feed-forward memory allocation for continual learning in neural machine translation. (Cit. on p. 47).

Xu, J., Zhou, H., Gan, C., Zheng, Z., & Li, L. (2020). VOLT: improving vocabularization via optimal transport for machine translation. *CoRR* (cit. on p. 26).

Xue, L., Barua, A., Constant, N., Al-Rfou, R., Narang, S., Kale, M., Roberts, A., & Raffel, C. (2021). Byt5: Towards a token-free future with pre-trained byte-to-byte models. *CoRR* (cit. on pp. 6, 26).

Yang, W., Ma, S., Lin, Y., & Wei, F. (2025). Towards thinking-optimal scaling of test-time compute for llm reasoning (cit. on p. 165).

Zhang, T., Kishore, V., Wu, F., Weinberger, K. Q., & Artzi, Y. (2019). Bertscore: Evaluating text generation with BERT. *CoRR* (cit. on pp. 36, 59).

Zhang, X., & LeCun, Y. (2017). Which encoding is the best for text classification in chinese, english, japanese and korean? *CoRR* (cit. on p. 75).

Zhang, Z., Tao, R., & Zhang, J. (2023). Neural network pruning by gradient descent. (Cit. on p. 149).

Zhu, W., Liu, H., Dong, Q., Xu, J., Huang, S., Kong, L., Chen, J., & Li, L. (2024). Multilingual machine translation with large language models: Empirical results and analysis. (Cit. on p. 48).

Zoph, B., Yuret, D., May, J., & Knight, K. (2016, November). Transfer learning for low-resource neural machine translation. In J. Su, K. Duh, & X. Carreras (Eds.), *Proceedings of the 2016 conference on empirical methods in natural language processing* (pp. 1568–1575). Association for Computational Linguistics. (Cit. on pp. 31, 39).