Departament de Sistemes Informàtics y Computació
Universitat Politècnica de València

A study on the impact of neural architectures for Unsupervised Machine Translation

MASTER'S DEGREE FINAL WORK

Master's Degree in Artificial Intelligence, Pattern Recognition and Digital Imaging

Author: Aitana Sanz Rodríguez
Tutor: Francisco Casacuberta Nolla

Year 2020-2021

Resum

La qüestió de l'ús de corpus monolingües per a l'entrenament de sistemes de traducció automàtica no supervisada es un assumpte de notable rellevància en aquest mon en contínua globalització en que vivim, a causa principalment de l'escassetat de corpus bilingües per a la gran majoria de parells de llengües i a les limitacions que açò presenta per l'entrenament de sistemes de traducció automàtica.

Aquest TFM pren com a punt de partida els sistemes de traducció neuronal no supervisada creats per Artetxe et al., anomenats Undreamt i Monoses, i aspira a explorar l'ús de diverses arquitectures neuronals properes a l'estat actual de la qüestió en el marc d'aquests sistemes.

S'utilitzaran per a açò diversos dels corpus monolingües provinents de la tasca de traducció WMT 2014, mesurant la qualitat de les traduccions aconseguides mitjançant la mètrica BLEU i buscant les millors configuracions per a diversos parells d'idiomes, comparant-les tant amb l'estat de la qüestió com a les mètriques reportades per Artetxe et al.

Paraules clau: Traducció Automàtica Neuronal, Aprenentatge No Supervisat, Transformer, LSTM, GRU

Resumen

La cuestión del uso de corpus monolingües para el entrenamiento de sistemas de traducción automática no supervisados es un asunto de notable relevancia en este mundo en continua globalización en que vivimos, debido principalmente a la escasez de corpus bilingües para la gran mayoría de pares de idiomas y a las limitaciones que esto presenta para el entrenamiento de sistemas de traducción automática.

Este TFM parte de los sistemas de traducción neuronal no supervisada creados por Artetxe et al. llamados Undreamt y Monoses, y aspira a explorar el uso de diversas arquitecturas neuronales cercanas al actual estado de la cuestión en el marco de dicho sistemas.

Se utilizarán para ello diversos de los corpus monolingües provenientes de la tarea de traducción WMT 2014, midiendo la calidad de las traducciones obtenidas mediante la métrica BLEU y buscando las mejores configuraciones para diversos pares de idiomas, comparándolas tanto como con el estado de la cuestión como con las métricas reportadas por Artetxe et al.

Palabras clave: Traducción Automática Neuronal, Aprendizaje No Supervisado, Transformer, LSTM, GRU

Abstract

The use of monolingual corpora for training Unsupervised Machine Translation systems is a matter of notorious relevance in this wold in continuous globalization we live in, mainly due to the scarcity of bilingual corpora for the great majority of language pairs and the serious limitation this represents for the training of Machine Translation systems.

This TFM takes as a starting point the unsupervised Neural Machine Translation systems created by Artetxe et al., named Undreamt and Monoses, and aims to explore, within the frame of said systems, the use of neural architectures that stand close to the current state of the art.

To do that the corpora used will be monolingual corpora from the WMT 2014 translation task, measuring the quality of the translations achieved using the BLEU metric and looking for the best configurations for various language pairs, comparing these both with the state of the art and with the metrics reported by Artetxe et al.

Key words: Neural Machine Translation, Unsupervised Learning; Transfomer, LSTM, GRU

Contents

List of Figures

List of Tables

CHAPTER 1
Introduction

This first chapter aims to introduce our work as well as the structure in which it will be presented, detailing our motivation and goals, the expected impact of our work and the methodology we will apply, and finally the structure this report will follow.

1.1 Motivation

It is nowadays undeniable that we live in a world in continuous globalization, a world in which people from all over the world interact with each other, in many cases, on a daily basis, and in which facilitating communication becomes a more relevant issue by the day.

Currently there are more than 7000 languages spoken worldwide [1], and only a very small percentage of them have bilingual parallel corpora publicly available. There are, of course, many more languages for which text corpora of different types can be found, yet they remain a very small subset overall of all the languages in the world.

Machine translation, as a discipline, investigates the use of software to translate between languages, yet both Statistical Machine Translation and Neural Machine Translation require large amounts of pre-processed text in the languages that are to be translated, as well as powerful machines that can train the necessary models. There is, however, a sub-discipline of machine translation that investigates the translation between languages without the use of parallel corpora, a sub-discipline known as Unsupervised Machine Translation.

It is Unsupervised Machine Translation, and particularly the State-of-the-Art discipline of unsupervised Neural Machine Translation, that could open many possibilities to the users of many languages for which parallel corpora are not easily available, yet for which is possible to obtain large corpora, as well as for the users of languages that while having large parallel corpora available still suffer from the limitation of too small corpora or the inability to obtain corpora between theirs and another language.

1.2 Goals

Given the means available to us at the time of developing this project, the goals pursued in this final Master's project are threefold:

- To successfully install and test two different frameworks for Unsupervised Machine Translation, the first being fully based on neural networks and the second being a hybrid from, statistical and Neural Machine Translation.

- To develop and improve upon those frameworks, and test different neural architectures within them.
- To compare the neural architectures used within these frameworks and find the best settings and parameters within our means.

1.3 Expected impact

We aim to present a comparison of the impact and effect of using different neural architectures within two different training frameworks, in a setup that is smaller and less powerful than what is reported, and both offer the best options and alternatives for training in similar setups and share our experience while developing this project, so that further investigations can advance beyond what this project has achieved.

1.4 Methodology

So as to properly present the comparison we aim for, this report will first offer an overview of machine translation, first as a discipline and afterwards focusing on its different sub-disciplines, before detailing the framework for the experiments developed and then the experiments themselves, divided between the two frameworks we have used, and then it will present the conclusions we have reached and those developments we propose for future works.

1.5 Structure of the Report

This Master's Degree Thesis is divided in chapters according to the following structure: this first chapter serves as an introduction to the overall intention of this report, having described the motivation, goals and expected impact of this report, as well as the methodology that will be used.

The second chapter further introduces both what Machine Translation is and what the state of the art is for this discipline, as well as any relevant concepts needed for a correct comprehension of this report. Furthermore and due to the relevance of Machine Translation with monolingual corpora, the third chapter goes into detail regarding both the relevant concepts and the state of the art for this particular sub-discipline.

The fourth chapter contains the experimental framework for this report, detailing the experimental setup available at the time of performing the experiments and writing the report, as well as the tools, frameworks, metrics and corpora used.

The fifth chapter details the performed experiments, including all the necessary comparisons with the reported experimental framework for the software and tools used in the experiments, as well as any modifications and the results obtained.

The sixth and last chapter of this report exposes the conclusions reached after performing all the detailed experiments, as well as what we propose as future work.

CHAPTER 2
Machine Translation

This chapter aims to introduce Machine Translation as a discipline, as well as describe the most important subdisciplines and their divisions, that is, exploring the differences and similarities between statistical and Neural Machine Translation, as well as those between supervised, semi supervised and Unsupervised Machine Translation, before briefly detailing the concept and significance of corpora and training data.

2.1 What is Machine Translation

It's in our nature, as humans, to communicate. As we evolved we developed languages, and those languages have affected each other, as years and centuries have gone by, to become those we know today and those that we have lost to time. And even as languages diverged even more, communication has never been any less important.

Translation, as a human discipline, has never been as relevant as it is in this world in continuous globalization we live in. But human translators are nevertheless human, and the cost in time and in money to translate everything is not always feasible. Machine Translation aims to complement that discipline, to offer models that when trained can translate from a language to another, sometimes in various directions and sometimes in one alone, and while current machine translation has not managed to equal a human for more complex tasks, there is much that can be advanced, much that can suddenly become available, with the correct developments in this discipline.

2.1.1. Subdisciplines of Machine Translation

For clarity and so as to offer a proper view of machine translation as a whole, we explore two different classifications for the subdivisions of this discipline: we will divide them between statistical and neural models and architectures, and between supervised, semi-supervised and Unsupervised Machine Translations.

Moreover, this chapter contains an overview of Statistical, Neural, Supervised and Semi-supervised Machine Translation, as well as their corresponding state of the art, while the next chapter offers a much more in depth view of Unsupervised Machine Translation, both Statistical and Neural, so as to properly represent the scope of this work.

Statistical and Neural Machine Translation

The differentiation between Statistical and Neural Machine Translation is one that is based on the architecture and design of the models: Statistical Machine Translation, older and developed first, bases its models on statistical models, against the neural networks that are the cornerstone and base of Neural Machine Translation.

There are, moreover, hybrid models, which will be detailed mainly in the following chapter due to their use in Unsupervised Machine Translation. These are, most often, models that train a neural network over a previously trained statistical model.

Supervised, Semi-supervised and Unsupervised Machine Translation

While the previous classification, that which divides between statistical and Neural Machine Translation, deals with the inner workings of the model or architecture, this classification refers to the data used for the training of said models.

While further detail will be provided in the corresponding sections, a task or work falls in the domain of Supervised Machine Translation when it is trained exclusively with parallel data, that is, data in two or more languages in which the sentences of the source corpus are a translation of those of the target corpus, and the other way around.

Unsupervised Machine Translation thus refers to models which are trained without the introduction of parallel data, with both corpora used being unrelated, whether they belong to the same domain or no. Consequently, Semi-supervised Machine Translation refers to the training of models with both parallel and non-parallel data.

2.1.2. Corpora and training data

For Machine Translation, and going forward for our experiments, corpora refers to the texts that are used to train the Machine Translation model. As detailed in the previous section, corpora can be parallel, that is, two corpus that contain the translation of each others' sentences in order, or not.

Training data refers to all the inputs used to train a Machine Translation model. Depending to the framework and the subdiscipline it can refer only to corpora or include the alignment vectors, or another kind of pre-processed material. This will be discussed in detail in the corresponding section of each framework, if it applies.

The effect of translated text in machine translation

The translation of texts into another language, whether it's done by a person or a machine, produces text that in general is thought to possess some particularities, as reported by Baker et al. [2], such as less ambiguity, major simplification, a preference for conventional grammaticality, among others. As detailed by Graham et al. [3], it's common to evaluate systems on a sample of human-translated text, and many test sets are comprised in large parts of translations, so as to create test sets for two directions simultaneously at no extra cost. This can, however, result on the worse performance of systems tested on these data, as the features pointed out for translated language are not observed as commonly in non-translated text.

2.2 Statistical Machine Translation

This section aims to briefly describe Statistical Machine Translation and the mechanisms that make this approach possible, touching the different architectural designs for statistical models, the challenges this subdiscipline faces and the current state of the art for Statistical Machine Translation.

2.2.1. Statistical models

The models for Statistical Machine Translation can be classified in word-based and phrase-based models, and in language models. We will go into brief detail for both, and furthermore introduce Moses, a Statistical Machine Translation framework that will be relevant for the experiments described in later chapters.

Word-based and phrase-based models

The first Statistical Machine Translation models based their mechanisms on considering words as atomic units, and the translation from sentence to sentence as a mapping between the words that form it in the source and target languages, as shown in 2.1. Due to often learning from parallel corpora, Statistical Machine Translation uses both a translation model and a language model, the latter modeling the alignments between words that the corpora provided for training, due to only providing the translation, do not offer.

Phrase-based Machine Translation is a later and more successful approach to Statistical Machine Translation, which divides the input into phrases and considers these phrases as atomic units instead of words, often obtaining these phrases from annotated parallel corpora or learning them directly from said corpora.

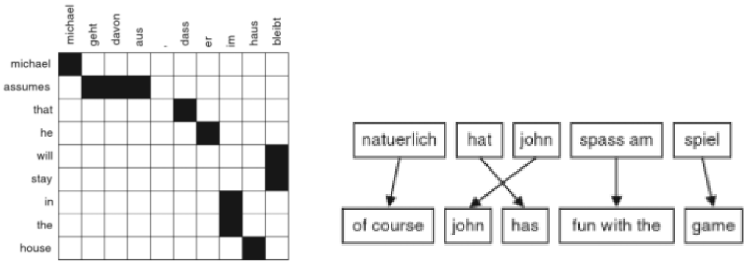

Figure 2.1: Word and phrase alignment for Statistical Machine Translation [4]

Language models

Language models are used to measure the fluency of the output [4], and thus are an essential component which influences word choice and reordering, among other choices. They are optimized on their perplexity and assign each sentence a probability, which is obtained via the product of the probability of each word, given each word's history.

As a part of the greater whole, the greatest challenge in these models is the handling of sparse data, as the fact that a word hasn't appeared doesn't necessarily mean that it won't appear.

Moses

Moses [1] is an open-source implementation to the Statistical approach to Machine translation, first started in 2005, and licensed under the GNU Lesser General Public License [2]. It is comprised of two main components, those being the training pipeline and the Decoder, the former being a collection of tools and steps mainly written in perl, with some using instead C++, and the latter being a C++ application that takes a model and a source sentence and will translate it into the target language.

[1] http://www.statmt.org/moses/

[2] http://www.gnu.org/licenses/lgpl-3.0.html

2.2.2. Challenges

From among the challenges presented by the translation between two or more languages, such as the translation of numbers or names, we will discuss some that have been relevant for Statistical Machine Translation.

Morphology

The differences in morphology come into play with languages that are inflected, for which it can be useful to translate the lemma and morphemes separately, and when dealing with compound words or languages that create words via the aggregation of other words, such as German.

Difference in syntactic structures

For those languages with different syntactic structure, when training statistical models, both word-based and phrase-based models tend to have difficulties with the increased amount of reordering that is necessary during the translation process.

Sparse data

As mentioned in a previous section, statistical models often face the necessity to deal with sparse data, that is, words or tokens that have not appeared during previous iterations of training and yet might appear very sporadically. The use of large quantities of data for training and vocabularies does help, yet it does not entirely address this problem.

2.2.3. State of the art

While it is generally accepted, at this point in time, that Neural Machine Translation has superceded Statistical Machine Translation, as the former is the newest technology and has shown to provide better results, there still remain some researchers occasionally focusing on Statistical and Hybrid Machine Translation systems, even though the latter will be discussed in the next chapter.

So as to provide a brief overview of the current state of Statistical Machine Translation, we have focused on comparisons between this subdiscipline and Neural Machine Translation, on newly developed models, on comparisons between statistical models, on architectural improvements, and finally on works regarding the use of Statistical Machine Translation and pivot languages for low resource scenarios.

Comparisons with Neural Machine Translation

Much of the focus on Statistical Machine Translation nowadays is on comparisons with Neural Machine Translation. One such comparison is the one published in 2021 by Benkova et al. [5], which focuses on comparing phrase-based Statistical Machine Translation systems and Neural Machine Translation systems for the Slovak and English language pair and using the translation direction from English to Slovak, reporting a better quality for the translation as well as statistically significant differences between both models, in favor of the most current Neural Machine Translation.

Newly developed models

Beyond comparisons, however, in 2021 as well Esan et al. [6] publish a paper detailing the development of a Statistical Machine Translation model, particularly a syntax-based model, for translation between English and Igbo, a language spoken by the Igbo people, a group from eastern Nigeria.

Their motivation for the development of this model was the semantic errors that occurred in existing Statistical Machine Translation, and their model, as reported, outperformed the previous state of the art of the models in both NIST and BLEU scores.

Improvements in parts of the architecture

Regarding current improvements to Statistical Machine Translation, moreover, in 2018 Banik et al. [7] publish a paper detailing their approach towards optimizing the time required for decoding in Statistical Machine Translation, claiming that one single set of parameters cannot fit the structure of all texts. To obtain the best parameters for each text and do so at runtime, moreover, they use a machine learning-based approach.

They report that their approach results in significant performance improvements both for decoding time and for translation accuracy, using in their experiments, moreover, low-resource datasets such as English-Hindi and Bengali-Hindi.

Comparison between statistical models

Taking a look at a different kind of comparison, in 2020 Brita Banitz publishes a paper [8] comparing the German translations of part of a particular book, *The Awful German Language*, by Mark Twain, when translated by Systran [3] and Google Translate [4], the former being a rule-based system and the latter being a statistical system based on a large bilingual corpus.

Banitz exposes, in the conclusions of her paper, that Google Translate, the statistical system, fared better in all the evaluation methods used even though the result text was grammatically evaluated as non-native German. It did, however, fare well in terms of fluency.

Statistical machine translation and the use of pivot languages for low resource settings

Regarding other venues of relatively current investigation for Statistical Machine Translation, in 2017 Ahmadnia et al. [9] publish a paper investigating the use the pivot language technique -that is, using a bridging language to increase the quality of the translation between two other languages- regarding the low-resource Persian-Spanish pair of languages, and using English as a bridge between them.

They do detail their use of phrase-level and sentence-level pivoting, as well as suggest a method to combine triangulation pivoting and a standard direct statistical model so as to obtain a better translation. They report, moreover, that the use of the pivot language technique allowed them to obtain better Statistical Machine Translation results for their selected pair.

[3] http://www.systranet.com/translate

[4] https://translate.google.com/

2.3 Neural Machine Translation

This section aims to offer a brief description of Neural Machine Translation and what it entails, describing therefore neural models and their components, the most common neural architectures and the most common challenges Neural Machine Translation faces nowadays, before diving into the state of the art and how those challenges are being approached.

2.3.1. Neural Models

Before going into detail on each of the common architectures, this section will describe some necessary concepts, these being that of word embeddings, that of the Encoder-Decoder approach, as well as Attention, and what word alignment, model training and search entail.

Word Embeddings

Much of Neural Machine Translation is based on the prediction of words via previous words, and so a way to represent said words becomes a clear necessity. The solution given is the concept of word embeddings: they are vector representations of context words, based on the idea that words that occur in similar contexts are semantically similar. Word Embeddings are used by models to carry out semantic inference and thus make predictions of upcoming words.

Encoder-Decoder approach and Attention

Most current neural architectures are based on an approach combining an Encoder, a Decoder, and an Attention mechanism.

The former, the Encoder, is a recurrent neural network which consults the embedding matrix so as to process the input sentence, which is a sequence of words, and then provide its representation, encoding each word with a context based on the preceding words.

The Decoder, which is a neural network as well, takes the representation of the input context given by the Decoder, as well as the previous hidden states and predictions, so as to obtain a prediction for the output words as well as a new hidden state. It's worth noting that it's usual for the Encoder and Decoder architectures to be of the same type.

Regarding the Attention mechanism, it computes the association between the given input words, as processed, and the hidden state of the Decoder, so as to produce a context state that represents the relevance of each input word to produce the next output word.

Training and Beam Search

The training of a Neural Machine Translation system often requires a high degree of parallelism, most often taking advantage of the computation abilities that GPUs offer and trying to optimize the process and the computation. Thus, it requires dividing a shuffled corpus (shuffled so as to avoid any biases) into batches, then divide those batches as necessary, often gathering together sentences of similar length. If there are many sentences that are not of equal or similar length, non-words are added to pad the length and then a mask is used to mark where the valid data ends.

Once those batches are processed, moreover, the obtained gradients are gathered and then applied to one of the large batches the corpus was initially divided into to update the parameters. The progress of the training is commonly checked by using a validation set that is not part of the training data, as neural networks tend to have a point after which the error on this particular set does not improve and performance might even become worse, this being a phenomenon named *overfitting*.

Regarding Beam Search, in Neural Machine Translation, at each step of the training, one output word is predicted. To decide on said prediction, given the probabilities of many worlds, a beam of the top most likely n words, scored by their probabilities, are kept and then used to make different word predictions for each, by accumulating the probabilities of the possible partial translations, word by word, until a sentence is completed, then removing said possible sentence from the beam until no more hypotheses remain. The hypothesis with the highest scoring is considered then the best translation, and thus chosen.

2.3.2. Neural Architectures

This section will discuss and detail the most commonly used neural architectures, these being Recurrent Neural Networks and their subsets, Gated Recurrent Units (GRU), Long Short-Term Memory units (LSTM) and the Transformer architecture.

Recurrent Neural Networks

As explained by Hochreiter et al. [10] in 1997, a recurrent neural network can be described as a neural network consisting of a hidden state **h** and an optional output **y**, operating on a sequence **x** of variable length. At each time step t, the hidden state $h_{(t)}$ of the network is updated by the function

$$h_{(t)} = f(h_{(t-1)}, x_t),$$

with f being a non-linear activation function. A RNN can, moreover, be trained to predict the next symbol of a sequence by learning a probability distribution over said sequence.

GRU

Gated Recurrent Units, first introduced by Kyunghyun Cho et al. [11] in 2014 are a gating mechanism in recurrent neural networks that is similar to a LSTM with a forget gate, yet lacking an output gate and consequently having fewer parameters.

As implemented in Pytorch [5], when applied to an input sequence each layer computes the following function for each element:

$$\begin{aligned} r_t &= \sigma(W_{ir}x_t + b_{ir} + W_{hr}ht - 1 + ghr) \\ z_t &= \sigma(W_{iz}x_t + b_{iz} + W_{hz}ht - 1 + ghz) \\ n_t &= tanh(W_{in}x_t + b_{in} + r_t * (W_{hn}h_{(t-1)} + b_{hn})) \\ h_t &= (1 - z_t) \odot n_t + z_t \odot h_{(t-1)} \end{aligned}$$

With h_t being the hidden state at time t, x_t being the input at time t, h_{t-1} being the hidden state of the layer at time $t - 1$ or the initial hidden state at time o, and r_t, z_t, n_t being the reset, update and new gates respectively. Moreover, σ is the sigmoid function and $\odot$ is the Hadamard product.

[5] https://pytorch.org/docs/stable/generated/torch.nn.GRU.html

The Pytorch implementation, moreover, allows for the creation of multilayer GRU recurrent neural networks, allowing for a probability of dropout.

LSTM

Similarly to Gated Recurrent Units, Long Short-Term Memory [10] are a recurrent neural network architecture that can process entire sequences of data due to having feedback connections, being comprised of a cell, an input gate, an output gate and a forget gate, the three gates being used to regulate the flow of information that goes into and out of the cell.

As implemented in Pytorch, when applied to an input sequence each layer computes, for each element, the following sequence:

$$
\begin{aligned}
i_t &= \sigma(W_{ii}x_t + b_{ii} + W_{hi}ht - 1 + bhi)\\
f_t &= \sigma(W_{if}x_t + b_{if} + W_{hf}ht - 1 + bhf)\\
g_t &= tanh(W_{ig}x_t + b_{ig} + W_{hg}ht - 1 + bhg)\\
o_t &= \sigma(W_{io}x_t + b_{io} + W_{ho}ht - 1 + bho)\\
c_t &= f_t \odot c_{t-1} + i_t \odot g_t\\
h_t &= o_t \odot tanh(c_t)
\end{aligned}
$$

With h_t being the hidden state at time t, x_t being the input at time t, h_{t-1} being the hidden state of the layer at time $t-1$ or the initial hidden state at time o, and i_t, f_t, g_t, o_t being the input, forget, cell and output gates respectively. Moreover, once again σ is the sigmoid function and $\odot$ is the Hadamard product.

Similarly to the GRU implementation, Pytorch allows for the creation of multilayer LSTM networks that also allow for a probability of dropout.

Transformer

The Transformer architecture, first described by Vaswani et al. [12] in 2017, is a neural architecture based on Attention mechanisms solely, contrasting with previous architectures that relied on recurrence and convolutions and quickly becoming the new state of the art in matters pertaining to machine translation.

The original Transformer follows an Encoder-Decoder architecture, using stacked self-Attention and connected layers for both the Encoder and Decoder, the former being comprised by six layers divided in a multi-head Attention mechanism and then a position-wise connected feed-forward network, and the latter being comprised of six layers as well, with those same sub-layers each and then a third sub-layer as well, which performs multi-head Attention over the output of the Encoder stack.

Both Encoder and Decoder employ residual connection around the sub-layers, and then layer normalization, as shown in figure 2.2.

2.3.3. Challenges

Despite how Neural Machine Translation has been the most promising machine translation approach for many years, it still faces many challenges in which it struggles. This section aims to offer a general overview of these challenges before exploring the state of the art and how those challenges are being tackled in recent years, as well as which improvements have taken place.

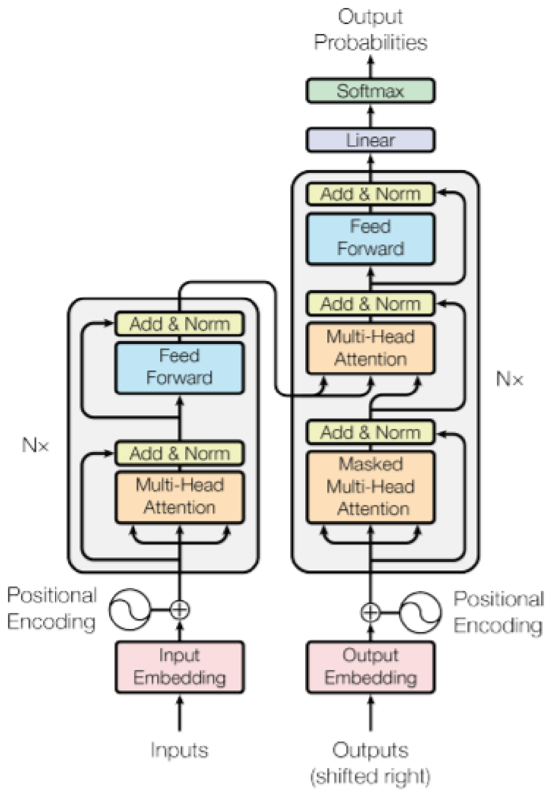

Figure 2.2: Transformer architecture as shown in the original paper. [12]

In the training data

There are many challenges that can arise due to circumstances and particularities present in the training data, particularly when compared to the target situations in which the model is going to be used. Among those, we can find the problems of Domain Mismatch, the quantities of training data, and the noise present in it.

Domain Mismatch refers to a circumstance in which the text to translate and the data in which the model has been trained belong to different domains, as words have different translations and different meanings depending on the general context. Thus, a model trained for a single context will encounter difficulties when translating text from another domain, despite how it isn't always feasible to train a model on the needed domain, most often due to the possible scarcity of training data.

Said scarcity of training data, while not a challenge for all domains and situations, remains a relevant one. The construction of a parallel corpus is expensive both in monetary and temporal terms, and even gathering monolingual data for training for some domains, and for many languages, is not always an easy task. Neural Machine Translation tends to exhibit a steeper learning curve depending on the training data available than Statistical Machine Translation [13], which further complicates matters. Moreover, the available data can exhibit various levels of noise that could affect the robustness of the model trained, such as misaligned sentences, the presence of other languages or even sentences translated in an incorrect way.

Word Alignment

In Neural Machine Translation, word alignment was first imposed by the addition of an Attention model and is obtained as a product of bidirectional gated recurrent neural networks. The challenge this presents when compared with traditional statistical word alignment methods is that the alignments seen in the Attention model states do not always match the statistical models, despite obtaining quality translations for both. [13].

Beam Search

Contrasting with Statistical Machine Translation, in which increasing the beam size parameter tends to result on better translations, in Neural Machine Translation this is not always the case [13], and thus looking for the best parameters becomes rather relevant challenge.

2.3.4. State of the Art

So as to offer a brief overview of the current State of the Art for Neural Machine Translation beyond unsupervised Neural Machine Translation, we have focused on three current subdisciplines: machine translation in low-resource settings, multilingual machine translation, and machine translation for sign languages.

Machine translation in low-resource settings

In 2019, Senrrich et al. [14] publish a paper discussing the validity of previous State-of-the-Art works regarding the situations in which Neural Machine Translation underperforms or not phrase-based Statistical Machine Translation, given the proven claims that the performance of Neural Machine Translation is severely hampered when in low-resource conditions. They proceed thus to discuss pitfalls from previous works, and to detail their experiments on German-English and different quantities of training data, with their NMT systems outperforming PBSMT systems with less resources than what previous works had claimed as necessary.

They proceed to discuss how in State-of-the-Art Unsupervised Machine Translation the NMT systems have, on many occasions, not been optimized for low-resource conditions, and proceed to suggest improvements in the methods for language representation, the tuning of hyperparameters and the lexical model used.

They then proceed to train both a baseline phrase-based statistical model as well as NMT models with their proposed improvement, observing an improvement on the BLEU results obtained for corpora of both 100k and 3.2M words.

Multilingual machine translation

In 2019, Aharoni et al. [16] publish a paper detailing their experiments in training a model that can translate up to 102 languages to and from English, by using training data composed of language pairs that contain English data either as the source or the target language. They use, particularly, both a corpus that contains parallel data in 58 languages and a corpus that contains 103 languages where either the source or the target is English. Their models, are, moreover, based on the Transformer model.

Aharoni et al proceed to train three models for their experiments with their corpora with parallel data for 59 languages, one of them being a many-to-many model with 58

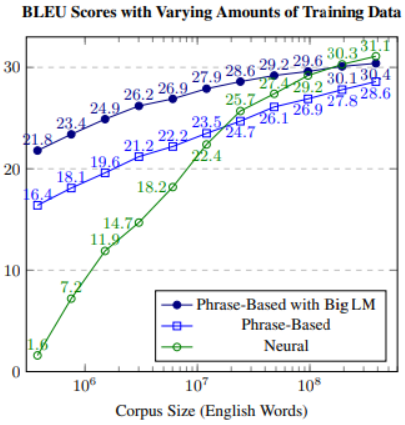

Figure 2.3: Quality of PBSMT and NMT in low-resource conditions according to Koehn and Knowles, 2017 [15]

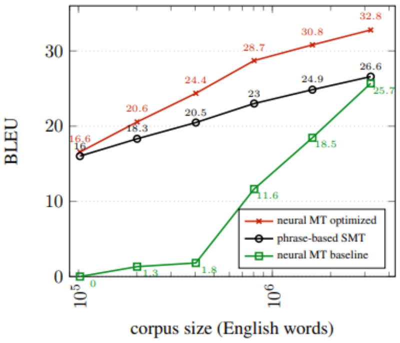

Figure 2.4: German→English learning curve, showing BLEU as a function of the amount of parallel training data, for PB-SMT and NMT, according to Sennrich et al, 2019 [14]

languages and thus 116 translation directions, and two many-to-one models, on from English to the rest of languages and another from those languages to English. Their models are comprised of six layers for both Decoder and Encoder, with model dimension 512, 8 Attention heads and a hidden dimension size of 2048. Their results show an improvement of more than 1 BLEU point over previous baselines.

Regarding their experiments for 103 languages, they train their models in 204 translation directions simultaneously, their model being a Transformer with 6 layers once again, model dimension of 1024, 16 Attention heads and a hidden dimension size of 8192. Their results show an improvement on the baselines in most cases, as well as their one-to-many model outperforming their many-to-many model in the translations from English to another target language.

Machine translation for sign languages

While this report deals mostly with matters of machine translation concerning written language, there is another subdiscipline of machine translation which is concerned with the translation of sign language, be it from sign to text or voice, or in the opposite direction.

In 2021, Farooq et al. [17] publish an article detailing the current approaches, limitations and challenges faced by this subdiscipline of machine translation, particularly Neural Machine Translation. They proceed then to perform an in-depth analysis of the current research, including the many angles it's currently being approached from: nowadays there are algorithms to translate natural language into sign language, as well as different approaches for sign recognition and creation, such as mobile applications or avatar generation.

The translation of sign languages is, moreover, a complex topic due to the very particular challenges it presents: there are hundreds of sign languages in the world, which do not share a grammar with the spoken language they share an area with, and thus the translation from spoken or writing language into signs is not as easy as it could seem. Moreover, the translation includes a visual medium which requires a particular encoding, unlike the usual textual encoding, in both directions, that most subdisciplines of Machine Translation deal with.

CHAPTER 3
Machine translation with monolingual corpora

This third chapter aims to detail the particularities of machine translation with monolingual corpora, including the circumstances that resulted in this particular subdiscipline of machine translation and the particular challenges it faces, as well as the current state of the art.

3.1 Unsupervised machine translation

While most architectures for Machine Translation, both statistic and neural, rely on parallel corpora, there is a subset of implementations that are attempting to train correct and capable Machine Translation systems that use corpora that are not necessarily parallel, that is, that the sentences provided do not only not contain the translations of their parallel corpus in the same order, but they may not contain them at all and may not even be in any way related to each other.

The reasons behind this approach are various, and most of them are easily visible. On the one hand, the creation and composition of parallel corpora is a lengthy, time-consuming process that requires a great economic investment, and such investments are mostly directed towards languages that either have a large amount of speakers throughout the world or are used commonly for business practices and international commerce. Those are, however, only a small subset of all the languages currently spoken in the world, a great percentage of which do not have any publicly available parallel corpora of any relevant size that relates to other languages, and if there are, the area they cover is both small and highly specialized.

Using monolingual corpora, however, for the purpose of training machine translation systems that achieve a significant quality, would make machine translation a much more available option for all the speakers of those languages. Suddenly the task of providing corpora large enough to train a machine translation system is no longer a matter of employing translators to ensure the quality of the parallel corpora used is up to the task, but a matter of compiling and organizing large amounts of texts such as news, articles, novels... which is, while not an easy endeavor, suddenly a more manageable and much less resource-intensive one.

3.1.1. Semi-supervised machine translation

While Unsupervised Machine Translation deals exclusively with monolingual corpora, there is a subdiscipline of Machine Translation, already introduced in our discussion of the different classifications of Machine Translation, that deals with both parallel and monolingual corpora.

The motivation for this discipline is not unlike that of Unsupervised Machine Translation: it aims to face the scarcity of resources for many language pairs by training systems with both parallel corpora and also monolingual corpora, aiming to enhance the performance of a model that would achieve far lesser results if trained with the or chosen parallel corpora by introducing large amounts of data for either or both the languages chosen as source and target.

3.2 State of the art

So as to offer a more in-depth overview of the current state of the art of Unsupervised Machine Translation, we have focused on Neural, Hybrid and Statistical Unsupervised Machine Translation, yet we have also taken a look at some models trained for particular translation directions, at investigations regarding word alignment in Unsupervised Machine Translation, at reports on the viability of Unsupervised Machine Translation, and finally on Multilingual Unsupervised Machine Translation as well.

3.2.1. Unsupervised Neural Machine Translation

In 2018, aiming to offer a solution to the problem that is the lack of large parallel corpora, Artetxe et al. [18] propose their method to train an unsupervised Neural Machine Translation System, using unsupervised embedding mappings as well as a modified Attentional Encoder-Decoder architecture, and being usable for Supervised and Semi-supervised training as well. Said architecture, moreover, is based on a shared Encoder for both translation directions, as well as on fixed cross-lingual embeddings.

It is too in 2018 that Lample et al. [19] propose a model that attempts to bridge the problem of the scarcity of parallel corpora by mapping the sentences of two different monolingual corpora and mapping them to the same latent space, the model then learning to reconstruct in both languages from this space and thus learning to translate without the use of any parallel data. Both Encoder and Decoder in this system encode and decode to this shared space, using a sequence-to-sequence model with Attention, the Encoder being a bidirectional-LSTM and the Decoder being a LSTM, both of them having three layers and sharing the Attention weights between the source and the target Decoder, and using greedy decoding.

This system proposed by Lample et al reports BLEU scores over 19 for the English-French and English-German language pairs, in both directions, the scores being noticeably higher for the English-French pair, and they are often used as baselines in further State-of-the-Art systems.

3.2.2. Hybrid and statistical unsupervised machine translation

In 2018 as well, and taking advantage of how Statistical Machine Translation is reported to obtain better results when the training dataset is smaller than what is usually demanded by Neural Machine Translation Systems, Artetxe et al. [20] propose their alternative approach to Statistical Machine Translation, further expanding upon it in 2019 [21], when they publish a paper detailing once more their approach and furthermore using it to initialize a dual Neural Machine Translation model.

Their reported approach is as follow: they build an initial phrase-table by using cross-lingual embedding mappings, which is later extended by incorporating subword information. Afterwards, the weights of the underlying log-linear model are adjusted through their unsupervised tuning procedure, and then the system is improved by jointly refining two models, one for each of the translation directions.

Once these models have been trained, they are used to assist the training of two unsupervised Neural Machine Translation systems, which are trained iteratively through single passes over a synthetic parallel corpus that is built by back-translation, said corpus being first generated by the SMT model and yet progressively being generated in a greater percentage, as the training progresses, by the reverse NMT model.

Once again in 2018, Lample et al. [22] publish a paper detailing two model variants for Unsupervised Machine Translation, one of them neural-network based and the other being a phrase-based model. They base themselves on the two aforementioned works, that of Artetxe et al. and that of Lample et al. so as to combine both neural approaches, obtaining a model that is reportedly easier to train and tune and outperforms the previous State-of-the-Art. They then proceed to apply those ideas and principles to a phrase-based Statistical Machine Translation model, and then to combine them. They proceed to then publish their scores for various language pairs, obtaining a clear improvement in the scores for most language pairs when using the combination of both models.

3.2.3. Models for only one or two translation directions

In 2019, Liu et al. [23] publish a paper detailing the *CAiRE* (Center for artificial Intelligence Research)'s submission for the Unsupervised Machine Translation track of the WMT 19 shared task, translating from German to Czech. The proposed system uses both word-level and subword-level Neural Machine Translation models, which are tuned by using pseudo-parallel data obtained from a phrase-based Statistical Machine Translation model. They do, moreover, train BPE embeddings for German and Czech separately, and then proceed to align those embeddings into a shared space by using the framework MUSE [24].

Furthermore, they train another model for the target language, this being Czech, which is then used to select and rescore the translation candidates that have been generated through beam search, before applying post-processing such as recasing and correcting named entities, so as to improve the quality of the translation.

3.2.4. Word alignment in Unsupervised Machine Translation

In 2018, Conneau et al. [25] present a paper discussing their method to obtain cross-lingual word embeddings without the use of parallel data. They claim that the previous methods for the creation of unsupervised word embeddings could not compete with supervised methods, while their method either equals or outperforms them, by using two large monolingual corpora and focusing on learning a mapping between two sets of embeddings trained independently on monolingual data. Their model is, moreover, based on Adversarial Training and reportedly offers proof of the feasibility of aligning word embedding spaces without the need for cross-lingual supervision.

Later yet in 2018 as well, Artetxe et al. [26] present a series of papers detailing and discussing their new method to create fully unsupervised cross-lingual mappings of word embeddings. They claim that previously proposed methods, in works such as the aforementioned paper by Conneau et al. [25], have been evaluated on particularly favorable conditions, while they wish to find a method that works for more realistic scenario.

They propose thus a method hat doesn't require the need of a seed dictionary, but instead focuses on the distributions of the words in the similarity matrix of all worlds in the vocabulary, claiming that two words that are the translation of each other in two different languages would have a similar distribution. They then proceed to share their results on various datasets containing pairs of English and another language, showing a general improvement.

3.2.5. The viability of Unsupervised Machine Translation

In 2020, Marchisio et al. [27] publish a paper discussing the conditions in which methods for Unsupervised Machine Translation succeed and fail, reporting that matters such as different domains and embedding training can dramatically affect the results of the training. They claim that recent successes in the field appear promising, yet the reported results shown are for languages for which traditional Machine Translation has already reported good results. Thus, they claim that for Unsupervised Machine Translation to be considered a viable path for low-resource situations, it must be determined if it works outside the highly-controlled environments in which it has been tested so far and furthermore how to evaluate promising training paradigms. Moreover, they expand on the former point: the methods must work when the languages are dissimilar or use different scripts, as well as between different domains, for target and source corpora or for training and test set, and with the low-quality data that is the reality for low-resource languages.

They report, furthermore, that the difference in domains makes the performance of the translation deteriorate, as does dissimilarity between the source and target languages, and that stochasticity during the training of the word embeddings can have a noticeable effect on the translation and the bilingual lexicon induction as well.

Moreover, they present an evaluation protocol for Unsupervised Machine Translation, so as to judge systems on dissimilar languages, on various showing a different degree of divergence between source and target corpora, on datasets that are diverse, and on actual low-resource language pairs.

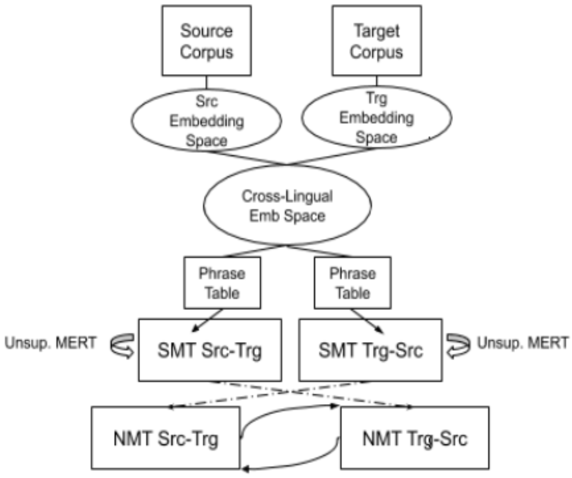

Figure 3.1: Unsupervised MT architecture used in the works of Marchisio et al. [27], based on the architectures used by Artetxe et al. [21][20]

3.2.6. Multilingual Unsupervised Machine Translation

In 2020, Garcia et al. [28] publish a paper discussing their approach to this particular subdiscipline of machine translation, discussing how the state of the art systems tend to perform poorly on languages with low resources available, citing how even the typically used languages to evaluate have comparatively great amounts of comparable data and how UNMT has been studied in mostly sterile setups.

Garcia et al. then proceed to detail their approach, adding a third stage to the existing two-stage models that consisted on pre-training with noisy reconstruction objectives as the first stage and then fine-tuning with back-translation and cross-translation. They propose then an intermediate stage that generates synthetic data to boost the accuracy, by leveraging offline back-translation. It must be noted that their setup uses auxiliary languages that contain both monolingual data as well as parallel data with English, as well as monolingual data for the target unsupervised languages. Their setup does then obtain BLEU scores that surpass those of the previous works they are compared against.

In 2020 as well, Garcia et al. [29] publish a paper detailing a probabilistic framework for multilingual Neural Machine Translation, which allows for both supervised and unsupervised setups. Their approach, moreover, is a focus on possible setups where there might exist parallel data between the source or target language and one other unrelated language, allowing for the use of high-resource language pairs so as to aid with the training of models for language pairs which do not offer as many resources. They do allow for other setups, including traditional unsupervised and supervised setups, as well as setups in which there is parallel data connecting the three languages.

They report BLEU scores approaching the state of the art for their models without auxiliary parallel data, as well as higher BLEU scores for the models they trained using it, in one occasion even surpassing previously reported supervised BLEU scores. They do, moreover, emphasize the impact of the chosen auxiliary language.

CHAPTER 4

Experimental Framework

In this chapter we will describe the setups and means with which we have performed the experiments, as well as the reported experimental setups of the frameworks we are working with, so as to allow for an easier comparison between them.

4.1 Intended goals

Within the hardware setup and the software frameworks that will be described in this chapter, the experimental goals of this report are twofold: we aim both to replicate the reported results of the two main frameworks that we will be using and to fine tune the parameters of their architectures for our hardware setup, which greatly differs, in some cases, from their reported experimental setup, so as to obtain the parameters and configuration that could allow for the training of functional Unsupervised Machine Translation systems in less powerful hardware setups than those originally reported.

4.2 Experimental setup

While the particular setup and model architecture for every experiment is further described in the corresponding chapter, just as each experiment is described, we proceed in this section to explain the hardware that has been used while performing the experiments that comprise the practical part of this report.

4.2.1. Hardware

During the development of this project we have majorly worked with two Nvidia GeForce GTX 1080 GPUs and a machine with 6 CPU cores and an available memory of 15.6 Gigabytes. For the training of the two Monoses Transformer models, moreover, we have worked with two Nvidia GeForce RTX 1080 GPUs.

4.3 Tools

This section aims to describe the frameworks and tools we have used during the development of our experiments, focusing both on the tools used for pre-processing and the frameworks used.

4.3.1. Pre-processing for Undreamt

Next we will describe the processes and tools we have used so as to prepare our chosen corpora for the first of the frameworks used in our experimentation, paying special Attention to the fact that we are working with corpora that are not parallel. It must be noted that the second of our frameworks, Monoses, already includes the pre-processing and requires only the raw text files.

Cleaning, lowercasing and tokenization

As we would do for any machine translation system, we first proceed to clean, lowercase and tokenize our chosen corpora. To do that, we will use the *clean-corpus-n*, *lowercase* and *tokenize perl* scripts available within the *Moses* [1] framework.

BPE

Byte Pair encoding, as applied to word segmentation and described in [30], is a data compression technique that iteratively merges frequent pairs of characters or character sequences into symbols, consequently obtaining functional representations of n-grams that can later be restored to the original tokenization while keeping the same vocabulary size of the original text.

Following Artetxe's recommendations to apply BPE for Undreamt [2] we use *subword-nmt* [3] to learn and apply Byte Pair Encoding to both the training and test corpora used.

Crosslingual embeddings

Following the application of Byte Pair Encoding, we proceed to train and map the embeddings for our corpora, with the due consideration to the fact that we are working with corpora that are not parallel, and thus training them with both languages so that they are automatically mapped to the same shared space is not an option.

Once again following Artetxe's reccomendations in [18], we use *word2vec* [4] to train monolingual embeddings on both sides of the pre-processed corpus with the settings described by Artetxe in [18], and afterwards we map those embeddings to a shared space by using *vecmap* [5], as described in [26].

4.3.2. Undreamt

Undreamt is a fully neural network based machine translation system we have used for some of our experiments. It is designed to be used on one GPU, and it's licensed under the GNU General Public License.

Original setup

As reported in [18], the training of the reported models took place in one *Titan X* GPU, and it required between 4 and 5 days for the training of each model.

[1] http://www.statmt.org/moses/
[2] https://github.com/artetxem/undreamt
[3] https://github.com/rsennrich/subword-nmt
[4] https://github.com/tmikolov/word2vec
[5] https://github.com/artetxem/vecmap

LSTM implementation

While the original Undreamt uses a GRU architecture, we have implemented a LSTM-based architecture by using the Pytorch LSTM implementation, which will allow us to compare both architectures.

Experimental considerations

Due to time and hardware constraints, in many of our experiments we have faced the need to reduce the size of our models, both for the GRU and for the LSTM architectures, particularly the latter due to its increased memory requirements.

4.3.3. Monoses

Monoses, created by Artetxe et al. [21], is a statistical and Neural Machine Translation hybrid framework for Unsupervised Machine Translation, which uses a Statistical Machine Translation system to initialize a dual Neural Machine Translation model that will be fine-tuned afterwards by using on-the-fly backtranslation. It's licensed under the GNU General Public License, and allows the training of a model to translate in both source-to-target and target-to-source directions at the same time.

Monoses trains the hybrid model throughout ten steps, which begin with corpus pre-processing, that is tokenizing, de-duplicating, cleaning by length and shuffling the corpora, as well as truecasing it, before splitting it in train and development sets.

The second step comprises the language model training via MOSES, before the third step which comprises training the embeddings to be used by extracting n-grams from the corpora, building a standard word2vec vocabulary and afterwards training the embeddings.

The fourth and fifth steps are respectively mapping the embeddings and inducing the phrase-table, before the sixth step which is building an initial model using, once again, Moses. The seventh step is then proceeding to tune the initial model, before proceeding in the eighth step to further expand the model via iterative backtranslation.

The ninth and final step before the neural hybridization consists on generating a synthetic parallel corpus, using both Moses and Subword NMT to learn BPE on the corpora, backtranslate them and apply BPE, and then extract the vocabulary from the corpora.

Finally, using as a basis the obtained model, the framework proceeds to train a Neural Machine Translation model, iteratively augmenting the weight of the Neural Machine Translation against the Statistical Machine Translation in as many iterations as passed by the parameter *Transition iterations*, and using Fairseq to train the Neural Machine Translation model.

Training is, moreover, reported to take one week in Artetxe et al.'s setup [6] by using 4 parallel GPUs.

Moses

So as to train the Statistical Machine Translation models that will later be hybridized with Neural Machine Translation, Monoses uses Moses, a Statistical Machine Translation framework which we have already detailed in the Statistical Machine Translation chapter, under the section Moses. More specifically, Monoses uses the Moses version v4.0.

[6] https://github.com/artetxem/Monoses

Fairseq

Fairseq [31] is a sequence modelling toolkit developed by Facebook AI Research, written in Pytorch and licensed under a MIT license, which allows for people to use it and modify it free of charge so long as the copyright and permission notice are respected. Monoses uses its version 0.6.

Its latest version is easily extensible, offering modular design and flexible configuration, as well as features such as mixed precision training, gradient accumulation and different implementations of Beam Search. The version used in Monoses no longer has documentation available, yet it offers various Transformer and LSTM models, among other architectures.

Other libraries

Beyond Moses, Monoses requires a subset of other libraries which we will list and briefly discuss in this section.

As previously mentioned, Monoses requires *Pytorch*, being reported as tested with the version v0.4, as well as *Java* and *Python 3*. Other libraries that it requires are *editdistance* [7], which implements the Levenshtein distance, *FastAlign* [32], a word aligner; *Phrase2Vec* [33], which is used to learn n-gram phrase embeddings; *VecMap* [26] which allows the build of cross-lingual embeddings without the need for parallel data, *Subword-NMT* [34], which offers preprocessing scripts for segmenting text into subword units; and *SacreBLEU* [35] which allows for computation of BLEU scores. Monoses' tuning module, moreover, is based on *Z-Mert* [8].

4.4 Metrics

4.4.1. BLEU

As a metric to measure the quality of the translations obtained via the experimental systems, we use BLEU, a language-independent automatic machine translation evaluation method first presented in 2002 [36].

4.5 Corpora

While we have used different corpora sizes for the experiments that will be described in the following sections, all of the corpora used for training are subsets of the *WMT14* [9] translation task, more specifically the News Crawl monolingual training data, obtained by concatenating the news crawl articles files from 2007 to 2013 for each of the languages involved in the experiments, so as to attempt to replicate the conditions stated by Artetxe et al in [18].

4.5.1. Languages used for the initial fine tuning process

For the experiments that have been performed with the explicit objective of fine tuning the parameters of our setup, we have chosen to use English and French, with the exper-

[7] https://github.com/roy-ht/editdistance

[8] http://cs.jhu.edu/ ozaidan/zmert/

[9] http://www.statmt.org/wmt14/

iments translating from French to English. This allowed us to gauge not only the BLEU metric but the quality of the obtained translation, as well as being a language pair that has been used and reported in the papers for both main frameworks we are using.

4.5.2. Languages used for further experimenting

So as to perform further experimenting with our setup, the corpora for one more languages has been added, this being German, and so we added the German to English translation direction to our experiments.

CHAPTER 5

Experiments and results

This chapter will focus on each of the experiments we performed in our attempt to compare Neural architectures within the Undreamt and Monoses Frameworks, with each experiment detailing our experimental setup, a comparison with the reported experimental setup if it is relevant, and the results obtained.

5.1 Recreating the previously reported results with Undreamt

In an attempt to test our experimental setup, we have attempted to recreate the experiments for the languages French to English reported by Artetxe et al. [18] within the means available to us, testing it with the original GRU.

5.1.1. Reported experimental setup

Artetxe et al. report a BLEU score of **15.56** when translating from French to English, either by using Backtranslation or BPE, and their reported setup is as described in Table 5.1.

5.1.2. Our experimental setup

Due to hardware limitations, and due to Undreamt being designed to be used on a single GPU, we found ourselves needing to reduce both the size of the network to 300 units per layer, as well as reducing the size of the embeddings to 200 instead of 300. In order to counter as much as possible the effects of this reduction, we increased the number of steps from 300.000 to 600.000, resulting in the setup described in table 5.2.

Embedding size:	300
Corpus size:	30 million sentences
Network layers:	2
Units per layer:	600
Steps:	300.000

Table 5.1: Original experimental settings for Undreamt

Embedding size:	200
Corpus size:	30 million sentences
Network layers:	2
Units per layer:	450
Steps:	600.000

Table 5.2: Our experimental settings to recreate the reported Undreamt results

5.1.3. Results

With the aforementioned experimental setup, we have obtained a BLEU value of **13.62** against the reported baseline of 15.56, a decrease we believe most likely caused by the hardware limitations of our setup.

5.2 Testing the Undreamt LSTM implementation

Having modified the Undreamt source code so that it uses a LSTM architecture instead of a GRU architecture, we proceed to test our implementation and compare it to the original architecture.

5.2.1. Experimental setup

Due to the larger memory requirements of the LSTM model, and in order to compare both implementations fairly, we have needed to reduce the size of the neural network used to 300 units, keeping nevertheless to 2 layers but training the model for thrice as many steps to a total of 900.000, and obtaining the settings described in table 5.3.

5.2.2. Comparison and results

After training both models for the aforementioned number of steps, the results obtained are compared with our baseline (despite the difference in hardware setups) in the figure 5.1. We can therefore see how the less complex GRU units outperform, within our setup, the LSTM units.

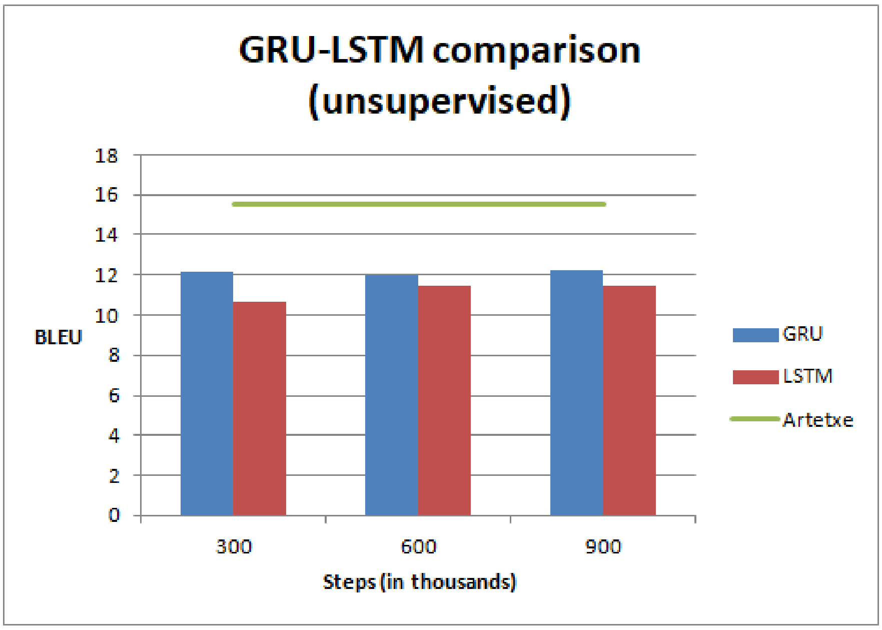

Figure 5.1: GRU and LSTM initial implementation comparison

Embedding size:	200
Corpus size:	30 million sentences
Network layers:	2
Units per layer:	300
Steps:	900.000

Table 5.3: Experimental setup to test and compare the GRU and LSTM implementations

5.3 Unsupervised tuning with the Undreamt GRU architecture

So as to obtain the best parameters for our settings, we proceed to to tune the Undreamt GRU architecture in search for the best Dropout, Embedding Size and Learning Rate Parameters.

5.3.1. Dropout and embedding size tuning

Using the largest network we are able to fit within the GPUs, and looking for the best Dropout and Embedding Size for our task, we proceed to train six different models, as explained in the following section.

Experimental setup

Given the minor memory requirements of the GRU implementation for Undreamt, we proceed to use it to test three different values for the parameter dropout, as well as testing them in two different embedding sizes, those being 200 and 300. Moreover, we perform the experiment in a network with one layer comprised of 500 units, this being the largest network we could fit and train in a GPU. The settings used for the experiment are, moreover, those shown in table 5.4.

Results obtained

As shown in figure 5.2, the best BLEU was obtained for the dropout value of 0.1, with embeddings of greater size performing better for larger dropout values, despite how the result for the dropout value 0,1 was slightly superior for embeddings of size 200. The time required to train each model, moreover, was approximately 40 hours.

5.3.2. Learning rate tuning

Following the experiment in which we found a value for the dropout parameter that better fit our settings, we proceed to test different values for the learning rate parameter, using once again the GRU implementation and the best found dropout value.

Experimental setup

For ease of comparison with previous experiments, we repeat the settings used for the first GRU experiments, these being networks with two layers and 450 units, as well as an embedding size of 200 due to hardware restrictions and a batch size of 50, resulting in the setup shown in table 5.5

Embedding sizes:	200, 300
Dropout values:	0.1, 0.3, 0.5
Corpus size:	5 million sentences
Batch size:	25
Network layers:	1
Units per layer:	500
Steps:	200.000

Table 5.4: Experimental settings for the unsupervised dropout tuning

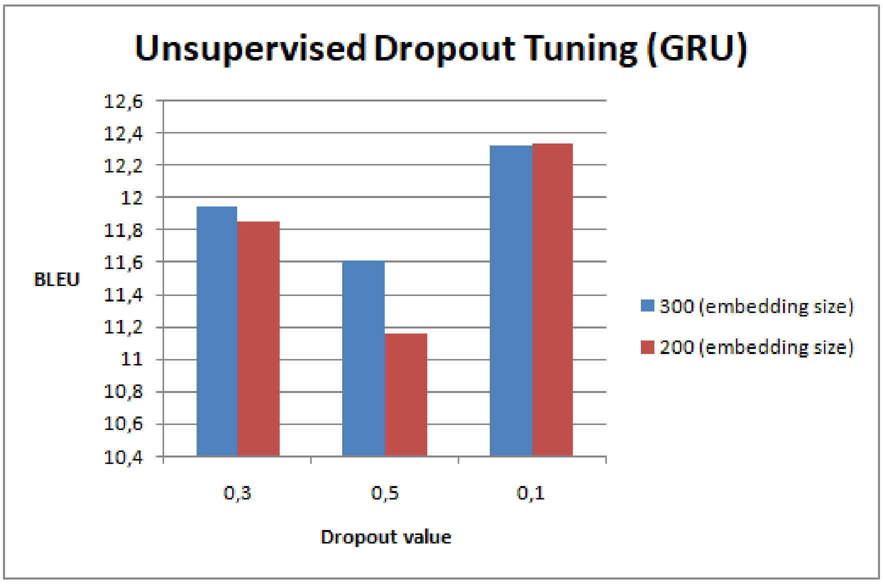

Figure 5.2: Unsupervised dropout tuning

Development

The first values we tried for the experiment were 0.002, 0.0001 and 0.0005, as the default learning rate value was 0.0002. Seeing how 0.0005 obtained better values than 0,0002, we tried the values 0.0004 and 0,0006. We did try the value 0.001 as well, although during the training this value presented errors in the calculation of the perplexity score and so was desestimated.

Results obtained

As shown in figure 5.3, the best value for the learning rate was obtained for the value 0.0004 as the learning rate, surpassing the default 0.0002 value.

Embedding sizes:	200
Dropout values:	0.1
Corpus size:	5 million sentences
Batch size:	50
Network layers:	2
Units per layer:	450
Learning rates:	0.002, 0.0001, 0.0002, 0.0004, 0.0005, 0.0006
Steps:	200.000

Table 5.5: Experimental setup for the unsupervised learning rate tuning

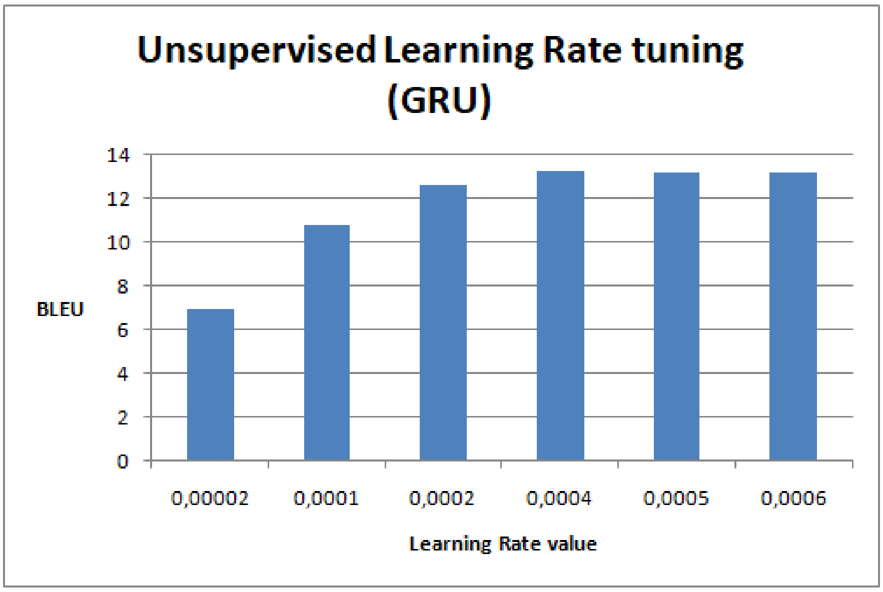

Figure 5.3: Unsupervised learning rate tuning

5.4 French-to-English Undreamt GRU and LSTM comparison

Having obtained the best parameters for our setup, both for the GRU and for the LSTM architecture, we proceed to compare both architectures by training a model with LSTM and a second model with GRU, both with the same size and layers and using the full corpus so as to obtain definitive results.

5.4.1. Experimental setup

Given the larger memory requirements for LSTM, we use the largest networks the GPUs can fit to train both models, using thus embeddings of size 300 and 280 cells per layer as described in the settings table 5.6, as well as 500.000 steps to make up for our smaller batch size.

We proceeded to train, thus, two different French to English models, with both architectures, obtaining results discussed in the next subsection.

5.4.2. Results

As shown in figure 5.4, the GRU architecture obtained slightly better values for all iterations, while requiring much less memory and approximately five hours less to finish

Embedding sizes:	300
Corpus size:	30 million sentences
Batch size:	50
Network layers:	2
Units per layer:	280
Learning rate:	0.0004
Dropout value:	0.1
Steps:	500.000

Table 5.6: Experimental setup for the French to English unsupervised Undreamt models

training for 500.000 steps, requiring approximately 3.5 days against the almost four days the LSTM model needed and thus appearing as objectively better for this task.

The best BLEU scores obtained were, moreover, **14.09** for the GRU architecture with 500.000 steps and **12.5** for the LSTM architecture with 300.000 steps, as shown as well in table 5.14

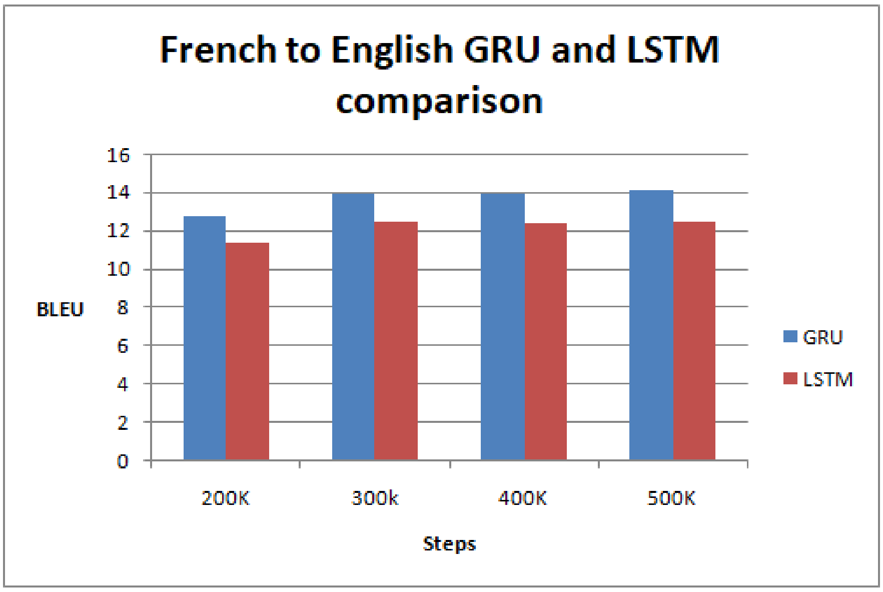

Figure 5.4: Comparison of GRU and LSTM French to English BLEU values

5.5 German-to-English Undreamt GRU and LSTM comparison

So as to test our Undreamt setup with a different architecture, we proceeded to train two models, from German to English, with similar settings to our French to English models. Due to time constraints, we used the default parameters

5.5.1. Experimental setup

So as to allow for a better comparison with our French to English models, we trained both a GRU and a LSTM model with the parameters described in table 5.7, which include the same number of layers and units per layer as the aforementioned models.

Embedding sizes:	300
Corpus size:	30 million sentences
Batch size:	50
Network layers:	2
Units per layer:	280
Learning rate:	0.0002
Dropout value:	0.3
Steps:	500.000

Table 5.7: Experimental setup for the German to English unsupervised Undreamt models

5.5.2. Results

As we can see in figure 5.5 , once again GRU obtained better BLEU values, from steps 300.000 onward, even though due to the lack of tuning the values obtained were notably inferior to those obtained in our French to English training.

The best BLEU scores obtained for these models were **7.26** for the GRU architecture with 500.000 steps and **7.06** for the LSTM architecture, with 300.000 steps, results shown as well is table 5.14

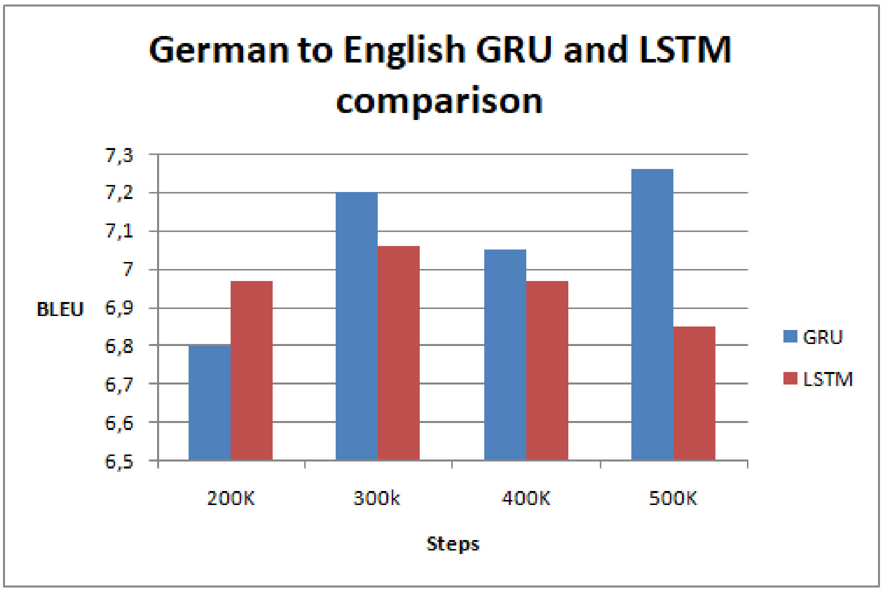

Figure 5.5: Comparison of GRU and LSTM German to English BLEU values

5.6 Testing our Monoses setup

So as to test our Monoses setup, given the even larger hardware differences between the reported setup and the means we have available, we proceed to train a model with this framework and a small LSTM model and ensure it works correctly.

Due to time and hardware constraints, moreover, particularly memory requirements, we used a smaller and less refined Statistical Model as well, and comparatively less transition iterations between SMT and NMT.

5.6.1. Reported experimental setup

Artetxe et al. report a BLEU score of **33.5** when translating from French to English, using the WMT14 corpora and a setup including 4 GPUs, as described in table 5.8.

5.6.2. Our experimental setup

Due to hardware and time constraints, we have chosen a LSTM model and reduced the size of the training corpus and the backtranslation corpus for both test and tuning, as well as the number of sentences per iteration, as shown in table 5.9. We have also reduced the backtranslation tuning iterations to 1 due to the large amounts of memory required for this step, and used instead only one iteration and thus the initial Moses weights.

Neural architecture model:	Transformer Vaswani WMT En De Big
Corpus size:	30 million sentences
Embedding size:	300
GPUs:	4
Max Tokens:	2500
Sentences per iteration:	1000000
Epochs:	60
Dropout:	0.3
Learning rate:	0.0005
Backtranslation tuning iterations:	3
Backtranslation training sentences:	10000000
Transition iterations between SMT and NMT:	30
Bitext generation sentences:	15000000

Table 5.8: Reported experimental settings for Monoses

Neural architecture model:	LSTM Luong WMT En De
Corpus size:	5 million sentences
Embedding size:	300
GPUs:	1
Max Tokens:	1250
Sentences per iteration:	200000
Epochs:	40
Dropout:	0.3
Learning rate:	0.0005
Backtranslation tuning iterations:	1
Backtranslation training sentences:	1000000
Transition iterations between SMT and NMT:	15
Bitext generation sentences:	1000000

Table 5.9: Our experimental settings for our Monoses test

5.6.3. Results

Our model obtained a BLEU of **8.37** for the French to English direction and **5.58** for the English to French direction, values much smaller than those reported for the framework, yet we used a much smaller corpus and proved that our Monoses setup was, at the very least, able to train a small model.

Due to the size and backtranslation iterations, moreover, the time required for the full training of this model was of approximately seven days, although it must be considered that Monoses allows for translation in both source to target and target to source direction.

5.7 Comparing neural architectures in the hybrid framework Monoses

Having tested our setup for Monoses, we now proceed to train a much larger statistical model and test three of the different neural architectures offered by Fairseq.

5.7.1. Our experimental setup

Due to hardware and time constraints, we have trained four models with the settings described in table 5.10 and the following three architectures, one of them being LSTM and two of them being transformer:

- Transformer Vaswani WMT En De big
- Transformer WMT En de
- LSTM Luong WMT En de

We have chosen the LSTM Luong model to better compare with our previous, much smaller, test model, and we have chosen the two transformer models both for their different sizes, the *WMT En De Transformer* being much smaller than the *Vaswani WMT En De Big*, and because the latter is the default Transformer architecture used in Monoses.

We have, moreover, due to temporal constraints, used the default hiperparameters for our models, as shown in the settings table, and the sizes for each of the three architectures are shown in tables 5.11 and 5.12.

Corpus size:	20 million sentences
Embedding size:	200
GPUs:	1
Max Tokens:	750
Sentences per iteration:	200000
Epochs:	60
Dropout:	0.3
Learning rate:	0.0005
Backtranslation tuning iterations:	1
Backtranslation training sentences:	1000000
Transition iterations between SMT and NMT:	10
Bitext generation sentences:	2000000

Table 5.10: Our experimental settings for our comparison of neural architectures with Monoses

Name:	LSTM Luong WMT En De
Encoder hidden size:	1000
Encoder layers:	4
Decoder hidden size:	1000
Decoder layers:	4

Table 5.11: Details of the LSTM architecture used with Monoses

Name:	Transformer WMT En De	Transformer Vaswani WMT En De Big
Encoder layers:	6	6
Encoder dimension:	2048	4096
Encoder Attention heads:	8	16
Decoder layers:	6	6
Decoder dimension:	2048	4096
Decoder Attention heads:	8	16

Table 5.12: Details of the Transformer architectures used with Monoses

Problems encountered during development

The original idea for this experiment was to test a LSTM and a Transformer model and then perform tuning over those until we could find hiperparameters that better fit our task.

Due to hardware constraints, particularly during the training of the statistical model, the process of finding a model small enough to make training feasible and yet large enough to offer quantifiable results made it so that time became our most severe constraint, and so we had to resort to the default hyperparameters, and, as the results showed, many less epochs than would have been necessary for the training of models as large as the Transformers.

5.7.2. Results

As we can see in figure 5.6 and table 5.13, two of our three architectures used for our test obtained notoriously lower BLEU values than our Undreamt and than our Monoses test, while requiring a much larger training time (that of six days for the Statistical model, adding six to seven days for each of the Neural models). The third architecture, that of the larger Transformer model, produced nonsensical translations with a BLEU score of 0, and so has not been included in the figure nor the table.

We believe, however, that with proper tuning and enough iterations, so as to perform various epochs over the data, a much larger BLEU score could be achieved, particularly in the case of the Transformer models.

Model	Translation Direction	BLEU
LSTM Luong WMT En De	French - English	3.6
LSTM Luong WMT En De	English - French	2.05
Transformer WMT En De	French - English	0.08
Transformer WMT En De	English - French	0.19

Table 5.13: Overview of Monoses models' translation scores

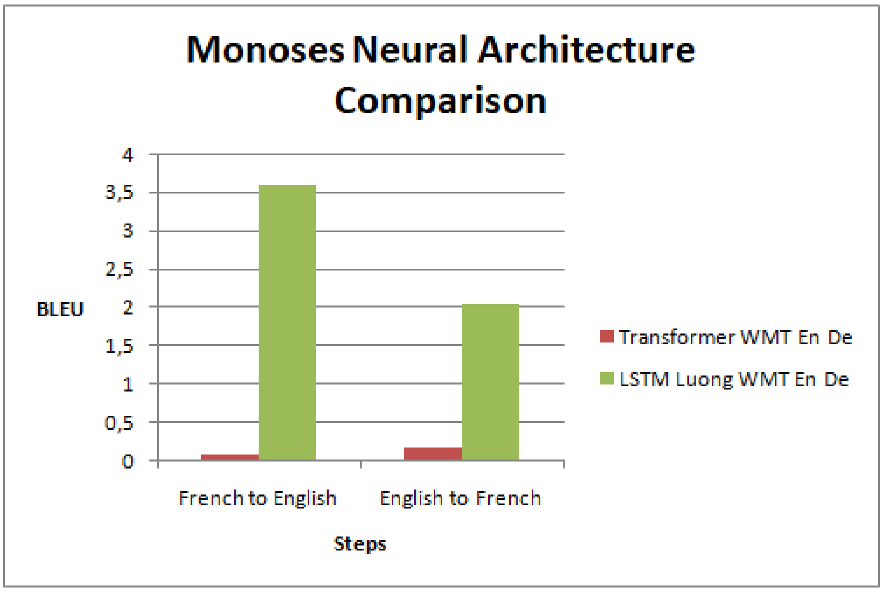

Figure 5.6: Comparison of Monoses models' translation scores

5.8 Quantitative overview of results

Having detailed all the experiments we performed, we proceed to detail the BLEU scores obtained by our final Undreamt and Monoses models, as shown in table 5.14, where our test Monoses model refers to our model trained with 5 million sentences, and the remaining Monoses models refer to those trained with 20 million sentences.

Framework	Model	Translation Direction	BLEU
Undreamt	GRU (500k steps)	French - English	14.09
Undreamt	LSTM (300k steps)	French - English	12.5
Undreamt	GRU (500 k steps)	German - English	7.26
Undreamt	LSTM (300 k steps)	German - English	7.06
Monoses	LSTM Luong WMT En De (test)	French - English	8.37
Monoses	LSTM Luong WMT En De	French - English	3.6
Monoses	Transformer WMT En De	French - English	0.08
Monoses	LSTM Luong WMT En De (test)	English - French	5.58
Monoses	LSTM Luong WMT En De	English - French	2.05
Monoses	Transformer WMT En De	English - French	0.19

Table 5.14: Overview of the scores obtained by our translation models

5.9 Qualitative comparison of translations

So as to observe the effect of the different architectures and frameworks, we now proceed to compare some of the translations obtained in our experiments, so as to offer a qualitative view of our results beyond the BLEU values obtained.

Tables 5.15 and 5.16 show sentences from our best Undreamt GRU model, from our test LSTM Monoses model and from our Transformer Monoses model.

Correlating with the BLEU values obtained by the translations of each of the models, the Undreamt model produced legible translations, the test LSTM Monoses model produced translations which did not conserve the proper meaning of the sentence (as seen in table 5.16) , and the Transformer Monoses produced nonsensical translations.

Reference:	The American Civil Liberties Union is deeply concerned, too, raising a variety of privacy issues.	
Framework:	**Model:**	**Sentence:**
Undreamt	GRU fr-en 500k steps	an american civil liberties union is also very concerned and expresses its concern regarding the protection of private life
Monoses	LSTM fr-en test model	The American Civil Liberties Union, which is also very concerned about its concern about the protection of privacy.
Monoses	Transformer fr-en model	"I'm not sure what I'm going to do," he said.

Table 5.15: Qualitative comparison of French to English translated sentences (1)

Reference:	There is going to be a change in how we pay these taxes.	
Framework:	**Model:**	**Sentence:**
Undreamt	GRU fr-en 500k steps	there will be the change in the way we pay these taxes .
Monoses	LSTM fr-en test model	There will be no change in the way we are paying these taxes. "
Monoses	Transformer fr-en model	"We're not going to be able to do that," he said.

Table 5.16: Qualitative comparison of French to English translated sentences (2)

CHAPTER 6
Conclusions

This final chapter aims to present the conclusions we reached during the development of this work, as well as describe all the improvements and expansions we propose as future work.

6.1 Conclusions

This work aimed to present a comparison of the effects of using different neural architectures within two different frameworks, particularly within harsher material and temporal constraints than those the frameworks had been developed for.

We have thus offered an overview of Machine Translation, going into detail about Machine Translation and particularly Unsupervised Machine Translation, and explored the two frameworks we aimed to use. Afterwards, we trained different models within our constraints, and then tested their translations.

6.1.1. Goals attained

Our goals were threefold: we aimed to install and test our two chosen frameworks, to develop and improve upon them, and to perform a comparison of neural architectures within those frameworks.

Our first goal, we met fully: we successfully installed both frameworks, Undreamt and Monoses, and tested their successful training of different translation models.

Our second and third goals, we attained partly: time and hardware constraints didn't allow us to test models as complex as we would have liked, and we were only able to test two neural architectures per framework. What we did not attain is explained and detailed in the following section, Future Work. We did, however, perform a comparison of different architectures, and obtained various clear results.

6.1.2. Conclusions reached

The conclusions we have reached are the following:

- Within the Undreamt framework, the much less complex GRU architecture was faster to train and obtained better results than the more complex LSTM architecture, despite the latter being more commonly used for larger projects.

- Within the Monoses framework, the less complex LSTM architecture performed better than the Transformer architecture, which depends much more thoroughly on the correct choice of hyperparameters and tuning, as well as requires more epochs for successful training, although the margin was relatively small.

Thus, while we are aware that Transformer is the current state of the art, and we believe that with dedicated tuning it could easily outpace any LSTM architecture, as it has been thoroughly proven, we nevertheless conclude that LSTM, and the more simple GRU, have shown to be more robust within our smaller setups.

6.2 Future work

This section addresses all the improvements and expansions that, due to matters of time or available means, have not been expanded upon in this Master's Final Work, as well as those expansions and improvements that the development of this report have brought to our attention, yet we could not act upon due to unfortunate time and hardware constraints.

6.2.1. Experiment replication and tuning

Replication of the experiments with more powerful hardware

One of the most decisive constraints the development of this report has faced, beside time, has been the hardware constraints of the GPUs used for the development of our experiments. It has required us resize the models and seek to tune them where it has been possible, while allowing us only an approximation of the tests we would have preferred to run.

Replicating our experiments within a larger setup, while less time consuming than their original development, would also probably allow for better tuning and better results overall.

Replication and tuning of our latter models

Due to time constraints, particularly due to the time required to train the larger Monoses models, we were not able to perform the tuning we wanted and aimed to perform over the Monoses Transformer models, nor over the German to English Undreamt models.

We believe that the BLEU we obtained would dramatically increase if enough tuning were performed over all models.

6.2.2. Framework upgrades

Upgrading Monoses and Undreamt to more recent Pytorch versions

For the development of the experiments described in this report, we have used the versions of Pytorch that the documentation of Undreamt and Monoses recommended, which were Pytorch v0.3 for Undreamt and Pytorch v0.4 for Monoses. We made attempts to modify Monoses and Undreamt to more recent versions of Pytorch, so as to facilitate modifications, yet due to the stark differences introduced by the versions immediately following both v0.3 and v0.4, time constraints had us relegate these improvements to Future Work.

Upgrading Monoses to more recent versions of Fairseq

Not unlike upgrading Monoses to a more recent version of Pytorch, we found ourselves unable to fully upgrade Monoses beyond the version it was tested to work on, which was Fairseq v0.6. Due to the scarce documentation available for this version of Fairseq, as well as the many models and architectures that became available and included in later versions of Fairseq, we believe this modification would be of great interest.

6.2.3. Framework modifications

Modification of Undreamt to implement Transformer architectures

One of the greatest hurdles that the experimentation for this report has faced, and one that once again time constraints didn't allow us to overcome, has been the modification of Undreamt so that it allows for architectures beyond GRU and LSTM, mainly due to the framework's non-modular coding and the use of a non State-of-the-Art version of Pytorch.

Adapting Undreamt so that it could implement Transformer architectures, be it the original Transformer or, preferably, any Pytorch-based model, is something that might probably require an entire overhaul and perhaps a whole rewrite of the code, yet it would adapt the fully neural Unsupervised Machine Translation system to the current state of the art and allow for further experimentation.

Modification of Undreamt to allow for parallel GPU usage

A notorious limitation of the Undreamt framework is how it is coded to use only one GPU. Yet another improvement upon it would be adapting it to parallel GPU training, facilitating thus the training of models that could be both larger and more complex, as well as requiring for less total time to complete the training.

Modification of Undreamt to allow the use of Checkpoints to continue training

While not entirely a limitation, during the writing of this report there have been a small number of instances in which external factors interrupted the training of an Undreamt experiment. This framework, not being designed to allow for continuation of the training process via a previous checkpoint, required us to restart the whole training process.

It also made deciding the scope of the experiments more complex, as we had to consider both the concern of over-training and thus wasting time, as well as the concern of not training for enough steps and thus wasting even more time as we trained the same model from scratch and for a higher number of steps. These are all concerns that would become negated by implementing continuation from chekpoints within this framework.

Bibliography

[1] G. F. S. Eberhard, David M. and C. D. F. (eds.), "Ethnologue languages of the world. twenty-fourth edition," 2021.

[2] M. Baker, G. Francis, and E. Tognini-Bonelli, *'Corpus Linguistics and Translation Studies: Implications and Applications'*. Netherlands: John Benjamins Publishing Company, 1993.

[3] Y. Graham, B. Haddow, and P. Koehn, "Statistical power and translationese in machine translation evaluation," in *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, (Online), pp. 72–81, Association for Computational Linguistics, Nov. 2020.

[4] P. Koehn, *Statistical Machine Translation*. Cambridge University Press, 2009.

[5] L. Benkova, D. Munkova, Benko, and M. Munk, "Evaluation of english–slovak neural and statistical machine translation," *Applied Sciences*, vol. 11, no. 7, 2021.

[6] E. Adebimpe and J. B. Oladosu, "Development of a syntax-based model for english-igbo statistical machine translation," *LAUTECH JOURNAL OF COMPUTING AND INFORMATICS*, vol. 2, no. 1, pp. 69–78, 2021.

[7] D. Banik, A. Ekbal, and P. Bhattacharyya, "Machine learning based optimized pruning approach for decoding in statistical machine translation," *IEEE Access*, vol. 7, pp. 1736–1751, 2019.

[8] B. Banitz, "Machine translation: a critical look at the performance of rule-based and statistical machine translation," *Cadernos de Tradução*, vol. 40, pp. 54–71, 2020.

[9] B. Ahmadnia, J. Serrano, and G. Haffari, "Persian-spanish low-resource statistical machine translation through english as pivot language.," in *RANLP*, pp. 24–30, 2017.

[10] S. Hochreiter and J. Schmidhuber, "Long short-term memory," *Neural Computation*, vol. 9, no. 8, p. 1735–1780, 1997.

[11] K. Cho, B. van Merrienboer, C. Gulcehre, D. Bahdanau, F. Bougares, H. Schwenk, and Y. Bengio, "Learning phrase representations using rnn encoder-decoder for statistical machine translation," 2014.

[12] A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. N. Gomez, L. Kaiser, and I. Polosukhin, "Attention is all you need," 2017.

[13] P. Koehn, "Neural machine translation," 2017.

[14] R. Sennrich and B. Zhang, "Revisiting low-resource neural machine translation: A case study," 2019.

[15] P. Koehn and R. Knowles, "Six challenges for neural machine translation," 2017.

[16] R. Aharoni, M. Johnson, and O. Firat, "Massively multilingual neural machine translation," 2019.

[17] U. Farooq, M. S. Rahim, N. Sabir, A. Hussain, and A. Abid, "Advances in machine translation for sign language: Approaches, limitations, and challenges," *Neural Computing and Applications*, 2021.

[18] M. Artetxe, G. Labaka, E. Agirre, and K. Cho, "Unsupervised neural machine translation," in *Proceedings of the Sixth International Conference on Learning Representations*, April 2018.

[19] G. Lample, A. Conneau, L. Denoyer, and M. Ranzato, "Unsupervised machine translation using monolingual corpora only," 2018.

[20] M. Artetxe, G. Labaka, and E. Agirre, "Unsupervised statistical machine translation," in *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, (Brussels, Belgium), pp. 3632–3642, Association for Computational Linguistics, November 2018.

[21] M. Artetxe, G. Labaka, and E. Agirre, "An effective approach to unsupervised machine translation," in *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, (Florence, Italy), pp. 194–203, Association for Computational Linguistics, July 2019.

[22] G. Lample, M. Ott, A. Conneau, L. Denoyer, and M. Ranzato, "Phrase-based neural unsupervised machine translation," 2018.

[23] Z. Liu, Y. Xu, G. I. Winata, and P. Fung, "Incorporating word and subword units in unsupervised machine translation using language model rescoring," 2019.

[24] A. Conneau, G. Lample, M. Ranzato, L. Denoyer, and H. Jégou, "Word translation without parallel data," 2018.

[25] A. Conneau, G. Lample, M. Ranzato, L. Denoyer, and H. Jégou, "Word translation without parallel data," 2018.

[26] M. Artetxe, G. Labaka, and E. Agirre, "A robust self-learning method for fully unsupervised cross-lingual mappings of word embeddings," in *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pp. 789–798, 2018.

[27] K. Marchisio, K. Duh, and P. Koehn, "When does unsupervised machine translation work?," 2020.

[28] X. Garcia, A. Siddhant, O. Firat, and A. P. Parikh, "Harnessing multilinguality in unsupervised machine translation for rare languages," 2021.

[29] X. Garcia, P. Foret, T. Sellam, and A. P. Parikh, "A multilingual view of unsupervised machine translation," 2020.

[30] R. Sennrich, B. Haddow, and A. Birch, "Neural machine translation of rare words with subword units," in *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, (Berlin, Germany), pp. 1715–1725, Association for Computational Linguistics, Aug. 2016.

[31] M. Ott, S. Edunov, A. Baevski, A. Fan, S. Gross, N. Ng, D. Grangier, and M. Auli, “fairseq: A fast, extensible toolkit for sequence modeling,” in *Proceedings of NAACL-HLT 2019: Demonstrations*, 2019.

[32] C. Dyer, V. Chahuneau, and N. A. Smith, “A simple, fast, and effective reparameterization of IBM model 2,” in *Proceedings of the 2013 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, (Atlanta, Georgia), pp. 644–648, Association for Computational Linguistics, June 2013.

[33] M. Artetxe, G. Labaka, and E. Agirre, “Unsupervised statistical machine translation,” in *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, (Brussels, Belgium), Association for Computational Linguistics, November 2018.

[34] R. Sennrich, B. Haddow, and A. Birch, “Neural machine translation of rare words with subword units,” 2016.

[35] M. Post, “A call for clarity in reporting BLEU scores,” in *Proceedings of the Third Conference on Machine Translation: Research Papers*, (Belgium, Brussels), pp. 186–191, Association for Computational Linguistics, Oct. 2018.

[36] K. Papineni, S. Roukos, T. Ward, and W.-J. Zhu, “Bleu,” *Proceedings of the 40th Annual Meeting on Association for Computational Linguistics - ACL ′02*, 2001.