Departamento de Sistemas Informáticos y Computación
Grupo de Reconocimiento de Formas y Tecnologías del Lenguaje Humano

UNIVERSITAT POLITÈCNICA DE VALÈNCIA

Estrategias de aprendizaje online de los pesos del modelo log-lineal en traducción automática interactiva

Autor: Mara Chinea Rios

Trabajo Final de Máster desarrollado dentro del máster
Inteligencia Artificial, Reconocimiento de Formas e Imagen Digital

Tutores: Prof. Francisco Casacuberta
Germán Sanchis-Trilles

Valencia, 2013

Índice general

Agradecimientos

Quisiera agradecer al doctor Francisco Casacuberta la oportunidad que me ha dado de colaborar con el grupo de investigación PRHLT, dándome la oportunidad de realizar este trabajo final de máster en un área tan interesante como es la traducción automática estadística. Agradecer sobretodo al doctor Germán Sanchis gracias por su paciencia, a sido imprescindible y mi guía a lo largo de todo el trabajo.
Un agradecimiento a mis padres por darme la oportunidad de crecer profesionalmente y apoyarme siempre a pesar de la distancia. Para concluir, quisiera darle las gracias a David por su apoyo y su cariño en los momentos más difíciles.

Resumen

La intervención de los traductores humanos en un escenario de post-edición para corregir las traducciones obtenidas a partir de los sistemas de traducción automática es aún muy necesaria para lograr la calidad deseada. El paradigma de la traducción automática interactiva (Interactive Machine Translation, IMT), es capaz de reducir el esfuerzo y tiempo que el traductor humano tiene que invertir en el proceso de corrección.

En este trabajo final de máster se plantea la utilización del paradigma de traducción automática interactiva, combinado con una aproximación que adecua los pesos del modelo log-lineal a cada una de las traducciones mediante diferentes algoritmos de aprendizaje online. Nuestro objetivo es que el sistema aprenda de los errores corregidos, favoreciendo la corrección de las próximas traducciones.

Para lograr lo anteriormente planteado se emplearon diferentes algoritmos de aprendizaje online: Discriminative Ridge Regression, Perceptron-Like y Passive Agressive, empleados estos en post-edición con resultados positivos. Para poder utilizar estos algoritmos dentro del escenario IMT fue necesaria una nueva formulación de cada uno de los algoritmos.

Con estas nuevas formulaciones, en este trabajo final de máster, se obtienen resultados diversos, dando la posibilidad de emplearse en nuevos planteamientos para lograr la calidad de las traducciones deseada y así disminuir el esfuerzo del traductor humano.

Abstract

In a post-edit scenario, the translations obtained by machine translator systems need to have been corrected by a human translator to obtain the desire quality. Interactive Machine Tranlator (IMT) paradigm is able to reduce the effort and the time that human translators have to invert in the correction process.

In this thesis, we propose to adapt the weights of the log-linear model in interactive machine translator. For adapting the weights of the log-linear model, we have utilizes different online learning algorithms. The main goal is that the system learns from the errors corrected. We propose to use three different online learning algorithms: Discriminative Ridge Regression, Passive Agressive and Percetron-Like. These algorithms has been used in post-edit scenario with good results.

These algorithms needed a new formulation in IMT sceneario. With these new formulations, we have obtained different results. These resuts give the posibility to use the new formulations to archieve the quality deseared and reduce efforts of the human translator in new problems.

Capítulo 1

Introducción

El lenguaje es un medio para los seres humanos de expresar sus emociones, pensamientos, sentimientos y para compartir su mundo con los demás. En el catálogo *Ethnologue : Languages of the World* [25] se tiene recogida información de un total de 7.105 lenguas vivas, lo que demuestra la diversidad y amplitud de los lenguajes. La comunicación entre individuos puede ser compleja si hablan diferentes idiomas y la solución para esto seria un lenguaje universal para todos, lo cual a pesar de sus beneficios tiene una gran cantidad de inconvenientes.

En la actualidad la cantidad de información generada es muy extensa y en muchos casos es necesaria en diferentes idiomas, como por ejemplo: noticias, información comercial (manuales, guías, etc,...) y la generada en organizaciones oficiales como las Naciones Unidas y el Parlamento Europeo.

La industria de la traducción se apoya fuertemente en los traductores humanos. La única desventaja con los traductores humanos es que son humanos y son un recurso limitado. El traductor humano puede tan solo traducir 2,500 palabras por día (unas 8 a 10 páginas), por lo que aparece la necesidad de automatizar el proceso de traducción con la ayuda de los computadores, surgiendo así la *Traducción Automática* [18](a la que nos referiremos a lo largo de este trabajo como Machine Translation, utilizando sus siglas en inglés MT). La traducción automática consiste en expresar en una lengua lo que está escrito o se ha expresado antes en otra de una forma automática, una máquina es la que realiza está acción. Durante los últimos años se ha tenido gran interés en desarrollar traductores automáticos cada vez con más precisión, para esto se han utilizado diferentes aproximaciones teniendo diferentes éxitos.

Uno de los primeros planteamientos para la MT fue la *traducción palabra a palabra*[47]. Este enfoque era el más sencillo e intuitivo y en él la traducción se realiza de la siguiente forma: se consultan las palabras en el idioma origen en el

diccionario y se generan directamente sus correspondencias en el idioma destino. Los problemas de esta metodología son evidentes: no se tiene en cuenta la estructura sintáctica de la frase, ni las relaciones semánticas que existen entre las frases, además de que no se asegura la formación correcta de la frase en el idioma destino. El método de *interlingua* [46] tiene como idea básica representar la frase de entrada en una lengua intermedia abstracta e independiente de las dos lenguas (entrada y salida) y posteriormente se traduce a la lengua destino. La traducción se realiza en dos fases: análisis del texto a traducir y generación del texto traducido.

La *Traducción Automática basada en el Reglas* [1] es la aproximación más tradicional en el campo de la MT. En ella el conocimiento lingüístico de los expertos humanos es transferido al sistema en forma de reglas adecuadas al problema. Esta aproximación aporta resultados aceptables sólo en dominios restringidos por lo que está lejano de alcanzar un sistema de traducción de propósito general. La causa de esto es el desarrollo y la acumulación de reglas en dominios grandes, por lo que se producen problemas de búsqueda a la hora de decidir cuál es la mejor regla a utilizar en cada caso.

Las técnicas anteriormente mencionadas no requerían de una base de datos con ejemplos de oraciones bilingües, en donde se tiene la oración tanto en el idioma origen como en el idioma destino al que queremos traducir. Con la aparición de corpus que contienen este tipo de oraciones, fue posible emplear otras metodologías en MT. Un sistema de traducción automática basado en corpus [15] toma como información de entrada un texto en un lenguaje origen y genera como salida la traducción correspondiente en un lenguaje destino utilizando como fuente de información principal gran cantidad de ejemplos de traducciones previamente realizadas. Existe dos fases en el desarrollo de un traductor basado en corpus:

- La fase de entrenamiento, que toma como partida gran cantidad de textos generados por traductores humanos. Estos textos han de ser preprocesados para facilitar su posterior manipulación y suele consistir en la detección de unidades de traducción, como palabras o frases. Los textos procesados son pasados a una etapa de entrenamiento, donde de forma automática se trata de extraer la información necesaria para realizar posteriores traducciones. Esta información es almacenada en una serie de modelos para su posterior utilización.

- La fase de traducción toma como partida un texto a traducir. Tratando de obtener la salida que se considere óptima combinando la información de los modelos obtenidos en la etapa de entrenamiento.

La *Traducción Basada en Ejemplos* [26] parte de la idea de realizar traducciones tratando de imitar ejemplos de sentencias similares de las que ya cono-

cemos su traducción. Para conseguir estos ejemplos de sentencias se utiliza un gran volumen de texto bilingüe convenientemente alineado. Todos estos ejemplos de traducciones han de ser almacenados en una base de datos de texto para que, con posterioridad, se facilite su localización a partir de cualquiera de las palabras contenidas en los ejemplos. En un programa de traducción basado en ejemplos, se introduce un texto de entrada del que se pretende realizar una traducción. El sistema fragmenta esta entrada en "unidades de texto" y busca en la base de datos los ejemplos almacenados que coinciden en mayor o menor medida con estas unidades. Explota los ejemplos encontrados combinándolos y produciendo una salida. [15]

La *Traducción Automática* mediante *técnicas estadísticas* fue planteada en la década de los 50 [47], donde se propone utilizar una aproximación empírica estadística. En los últimos años las técnicas estadísticas han experimentado un interés creciente. Este resurgimiento puede deberse a varias causas:

- Por un lado el creciente número de textos bilingües masivos de fácil disponibilidad (corpus de entrenamiento).
- Por otro los grandes avances en el mundo de los ordenadores nos permiten disponer de máquinas cada vez más rápidas y con mayor capacidad.

Uno de los principales responsables del éxito de las técnicas estadísticas es el centro de investigación IBM J. Thomas Watson [5]. Su trabajo se basa en el modelo de canal ruidoso de Shannon.

Los sistemas de*Traducción Automática Estadística* (al que nos referiremos a lo largo de este trabajo como Statistical Machine Translation, utilizando sus siglas SMT) en los que se basa el estado del arte tienen un gran potencial, aunque no brindan traducciones de alta calidad por lo que hay muchos investigadores trabajando en mejorar el estado del arte de SMT.

1.1. Traducción Automática Estadística

Estos sistemas le fueron ganando terreno a los sistemas basados en reglas por la flexibilidad que brindan para adaptarse a nuevos problemas, llegando a ser los más utilizados [8].

La aproximación de reconocimiento de formas a la traducción automática fue planteada en [4], en donde dada una oración a traducir $\mathbf{x} = x_1...x_J$, en un idioma origen, se pretende encontrar la oración $\mathbf{y} = y_1...y_I$ en un idioma destino que maximice la probabilidad a posteriori, representada en la siguiente ecuación:

$$\hat{\mathbf{y}} = \underset{\mathbf{y}}{\operatorname{argmax}}\, Pr(\mathbf{y}|\mathbf{x}) \tag{1.1}$$

Este planteamiento requiere encontrar la oración $\hat{\mathbf{y}}$ con mayor probabilidad dada una oración de entrada $\mathbf{x}$, por lo que previamente necesitaremos calcular

la probabilidad conjunta para todos los pares de oraciones $(\mathbf{x}, \mathbf{y})$. Este cálculo de probabilidad es casi imposible ya que no disponemos de tal cantidad de corpus bilingües, y por lo tanto debemos emplear *la regla de Bayes* para abordar el problema:

$$\hat{\mathbf{y}} = \underset{\mathbf{y}}{\text{argmax}} \frac{Pr(\mathbf{y}) \cdot Pr(\mathbf{x}|\mathbf{y})}{Pr(\mathbf{x})} \tag{1.2}$$

Siendo la maximización en función de $\mathbf{y}$ podemos omitir el denominador, obteniendo la siguiente fórmula:

$$\hat{\mathbf{y}} = \underset{\mathbf{y}}{\text{argmax}}\, Pr(\mathbf{y}) \cdot Pr(\mathbf{x}|\mathbf{y}) \tag{1.3}$$

Por tanto, se descompone $Pr(\mathbf{y}|\mathbf{x})$ en dos probabilidades diferentes. El *modelo de lenguaje estadístico* para el idioma destino $Pr(\mathbf{y})$, y el modelo de traducción inverso $Pr(\mathbf{x}|\mathbf{y})$. Mientras que el modelo de traducción $Pr(\mathbf{x}|\mathbf{y})$ capturará la relación entre palabras o segmentos de palabras entre el idioma desde el que estamos traduciendo hasta el idioma destino, el modelo de lenguaje $Pr(\mathbf{y})$, construido a partir de un corpus monolingüe, se asegurará de que la oración resultante a partir la oración de entrada $\mathbf{x}$ esté bien formada según el idioma al que estamos traduciendo.

Normalmente el modelo del lenguaje está basado en *n-gramas*[40]. Un *n-grama* es una subcadena de *n* elementos, en este caso palabras. Si *n*=1 es un unigrama, *n*=2 es un bigrama, etc. Se define el modelo del lenguaje de la siguiente manera:

$$Pr(\mathbf{w}) = \prod_{i=1}^{|\mathbf{w}|} Pr(w_i|w_1^{i-1}) \approx \prod_{i=1}^{|\mathbf{w}|} P_{r_n}(w_i|w_{i-n+1}^{i-1}) \tag{1.4}$$

donde **w** es la oración, $|\mathbf{w}|$ representa la talla de la oración **w**, w_i representa la palabra *i-esima* del n-grama, y w_1^{i-1} y w_{i-n+1}^{i-1} son una secuencia de palabras dentro del n-grama.

Para modelar adecuadamente la probabilidad $Pr(\mathbf{x}|\mathbf{y})$ se propusieron varios modelos. En [5] se definieron los modelos de alineamiento de palabras conocidos como modelos de IBM, donde la correspondencia entre la oración origen y su traducción se establece mediante variables de alineamiento ocultas.

1.1.1. Modelos de alineamiento palabra a palabra

Los modelos de alineamientos de palabras fueron introducidos a principio de la década de los noventa por un grupo de investigadores del centro de investigación de IBM [3]. Los cinco modelos propuestos por este grupo son conocidos como modelos de alineamiento, y son utilizados hoy día como base de referencia en el resto de investigaciones en este campo. La característica más importante de estos modelos es la introducción del concepto de alineamiento.

Este planteamiento parte de la idea de que cada palabra, o grupo de palabras de la frase de entrada, genera una o varias palabras en la frase de salida. En los modelos [3] proponen utilizar concretamente la siguiente definición: Un alineamiento oculto a = $a_1 \dots a_{|J|}$ describe una relación entre cada una de las posiciones origen *j* con una posición destino a_j. También puede contener alineamientos con $a_j = 0$ si se considera que la palabra origen *j* no se relaciona con ninguna palabra destino. De esta manera, cada una de las palabras origen pueden ser colocada con una única palabra destino, o con ninguna, mientras que las palabras destino pueden estar alineadas con ninguna, una o varias de la frase origen. Los modelos de IBM indican cómo calcular $Pr(\mathbf{y}|\mathbf{x})$ mediante la siguiente ecuación:

$$Pr(\mathbf{y}|\mathbf{x}) = \sum_{\mathbf{a} \in A(\mathbf{x},\mathbf{y})} Pr(\mathbf{y}, \mathbf{a}|\mathbf{x}) \tag{1.5}$$

donde $A(\mathbf{x}, \mathbf{y})$ se define como el conjunto de todos los posibles alineamientos de $\mathbf{x}$ a $\mathbf{y}$.

1.1.2. Modelo Log-Lineal

Existen diferentes autores que han abordado directamente el modelado de la probabilidad a posteriori $Pr(\mathbf{y}|\mathbf{x})$, entre los que podemos encontrar [34], [29]. Estos autores proponen en sus trabajos el uso del llamado *modelo log-lineal* para combinar los diferentes modelos de traducción, distorsión o reordenamiento y de lenguaje. Por tanto el *modelo log-lineal* se definiría de la siguiente forma:

$$Pr(\mathbf{y}|\mathbf{x}) = \frac{\exp \sum_{m=1}^{M} \lambda_m h_m(\mathbf{x}, \mathbf{y})}{\sum_{\mathbf{y}'} \exp \sum_{m=1}^{M} \lambda_m h_m(\mathbf{x}, \mathbf{y}')} \tag{1.6}$$

donde la regla de decisión viene dada por la expresión:

$$\hat{\mathbf{y}} = \underset{\mathbf{y}}{\operatorname{argmax}} \sum_{m=1}^{M} \lambda_m h_m(\mathbf{x}, \mathbf{y}) \tag{1.7}$$

$h_m(\mathbf{x}, \mathbf{y})$ es la puntuación de una función que representa una característica importante en la traducción de $\mathbf{x}$ a $\mathbf{y}$, *M* es el número de modelos o características y λ_m representa un peso de la combinación log-lineal. Típicamente $h(\cdot|\cdot)$ se calcula empleando un conjunto de entrenamiento y los valores λ son ajustados mediante un conjunto de desarrollo, más conocido por tuning (nombre en ingles).
La Ecuación 1.7 puede ser representada de forma vectorial empleando el producto interno, siendo más compacta:

$$\hat{\mathbf{y}} = \underset{\mathbf{y}}{\operatorname{argmax}} \sum_{m=1}^{M} \lambda_m h_m(\mathbf{x}, \mathbf{y}) = \underset{\mathbf{y}}{\operatorname{argmax}}\, \boldsymbol{\lambda} \cdot \mathbf{h}(\mathbf{x}, \mathbf{y}) = \underset{\mathbf{y}}{\operatorname{argmax}}\; s(\mathbf{x}, \mathbf{y}) \tag{1.8}$$

en donde $s(\mathbf{x}, \mathbf{y})$ representa la puntuación de la hipótesis $\mathbf{y}$ dada una oración de entrada $\mathbf{x}$.

Ajustes iniciales del Modelo Log-Lineal

Cuando el sistema recibe una frase de entrada genera una hipótesis que va a recibir una combinación de los modelos log-lineales que habitualmente son una combinación de modelos logarítmicos $h_m(\mathbf{x}, \mathbf{y})$. Dichos modelos no tienen la misma importancia en la decisión global por lo que es necesario ajustarlos según su influencia.

El propósito de los factores de escalado λ_m es ajustar el poder discriminante de su correspondiente función de características $h_m(\mathbf{x}, \mathbf{y})$. Este ajuste se realiza típicamente con un conjunto de desarrollo que contiene frases bilingües de los mismos idiomas a traducir.

El propósito es seleccionar la mejor combinación para estos pesos de forma que el error sea el mínimo al utilizar el conjunto de desarrollo, a partir de un métrica de calidad. Para ello se emplea la técnica llamada minimum error rate training (MERT) [28]. La métrica mas comúnmente empleada es *BLEU*[33], aunque MERT está ideado para optimizar cualquier métrica de calidad.

El método de ajuste más empleado para un sistema SMT consiste en traducir las oraciones del idioma origen del conjunto de desarrollo y producir un conjunto de N mejores hipótesis (llamado N-best) para la traducción de cada oración. A continuación, utilizando un algoritmo de optimización como el algoritmo propuesto por Powell[35], los valores para los factores de escalado son estimados de forma que las hipótesis con mayor puntuación dentro de la lista N-best son colocadas en la parte superior de la lista. Inmediatamente después, se traduce de nuevo el conjunto de desarrollo empleando los nuevos factores de escalado obtenidos en la iteración anterior y produciendo una nueva lista N-best, combinándola con la anterior. Este proceso se repite de forma iterativa hasta que la lista N-best no varía de una iteración a otra, y logrando obtener una lista de posibles traducciones para las oraciones del conjunto de desarrollo.

1.2. Traducción Automática Estadística basada en secuencias de palabras

Los modelos basados en *segmentos o secuencias de palabras* [43] [22] [49] [48], comúnmente conocidos como *phrase-based models*, se introdujeron con el objetivo de tener en cuenta la información del contexto, problema que se producía al utilizar la traducción *palabra por palabra*. Los modelos basados en *secuencias de palabras* dividen la oración de entrada $\mathbf{x}$ en bloques de secuencias de palabras, denominados *segmentos*, a diferencia de los modelos basados en palabras que utilizan como unidad básica una sola palabra. Los modelos de secuencias de palabras provienen del concepto de *segmentación bilingüe* que consiste en, segmentar las oraciones de origen y destino $\mathbf{x}$ e $\mathbf{y}$ en secuencias de palabras. Estos modelos aprenden la probabilidad de que una secuencia

contigua de palabras de entrada, $\tilde{x}_k \in \mathbf{x}$, se traduzca por otra secuencia de palabras de salida $\tilde{y}_k \in \mathbf{y}$. De esta forma el último paso es reordenar estos segmentos de salida para obtener la oración $\mathbf{y}$ como traducción a la frase de entrada dada. Los diccionarios estadísticos de segmentos bilingües sustituyeron los diccionarios estadísticos de pares de palabras. En esta segmentación sólo se consideran segmentos de palabras contiguas y no puede haber solapamiento entre ellos. Por tanto, el número de segmentos de la oración origen y el de la oración destino deben ser iguales, y cada segmento de origen se alinea únicamente con un segmento de destino y viceversa. Cuando se incluye el modelo basado en segmentos en el modelo log-lineal la calidad de un sistema de SMT mejora.

Existe una gran variedad de técnicas desarrollas para obtener el *modelo basado de secuencias de palabras*. Entre ellas se encuentran [43] [22], siendo la más utilizada la desarrollada por [49]. Esta técnica plantea que todos los segmentos que coinciden con el alineamiento de palabras son extraídos, utilizando en muchos casos los alineamientos de IBM descritos en la subsección 1.1.1. Para ello se utiliza la *simetrización* que consiste en combinar heurísticamente los alineamientos origen-a-destino y destino-a-origen, aprovechando que los alineamientos son muy restrictivos debido a que cada palabra destino se asigna únicamente a cero o a una única palabra origen. Una vez hecho esto, el conjunto de segmentos consistentes con los alineamientos de palabras simetrizados se extraen para cada par de oraciones del conjunto de entrenamiento. Se puede ver un ejemplo de este procedimiento en la Figura 1.1

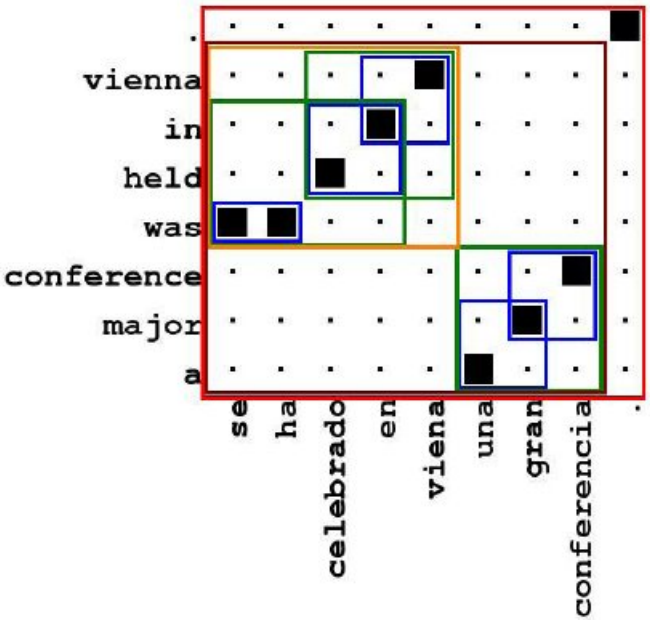


Figura 1.1: Ejemplo de extracción de segmentos basándose en alineamientos de palabras

1.3. Traducción Asistida por Ordenador

Los sistemas de traducción automática actuales nos brindan una idea global del contenido del texto, habiendo muchas tareas (ej: documentos legales, noticias, etc,..) donde estas traducciones no poseen la calidad necesaria y es necesario la colaboración de los traductores humanos para garantizar la calidad. La máquina es utilizada como una herramienta de trabajo por el traductor, por lo que la interacción hombre-máquina es cada vez más necesaria, dando lugar a diferentes métodos dentro del marco de la Traducción Asistida por Ordenador (a la que nos referiremos a lo largo de este trabajo como Computer-Assisted Translation, utilizando sus siglas en inglés CAT). El ordenador se convierte en una estación de trabajo a través de la cual el traductor humano tiene acceso a una gran variedad de recursos: por ejemplo diccionarios monolingües y bilingües, textos paralelos, textos traducidos en una variedad de lenguas y bases de datos terminológicas. Cada traductor puede crear su propio entorno de trabajo personal y transformarlo para ajustarse a las necesidades de cada tarea. Por tanto con CAT el traductor obtiene acceso a una impresionante cantidad de información actualizada. El resultado es un ahorro de tiempo, pero para que esto sea posible el papel de los ordenadores es muy importante.

La forma más sencilla de CAT se denomina post-edición y consiste en introducir la máquina al comienzo de un proceso secuencial [6], de forma que el sistema SMT genere una traducción temporal que el traductor humano se encarga de aceptar, rechazar o corregir. La calidad de la traducción inicial brindada

por el sistema SMT va a ser importante para la aceptación de este paradigma de post edición: cuanto mejor sea la calidad de las traducciones del sistema SMT, menores serán los cambios a realizar por el traductor, aumentado su productividad.

Existen otras formas más sofisticadas de introducir una máquina en el proceso de traducción. Entre ellas se encuentra el paradigma de la Traducción Automática Interactiva [2] [9] que combina el conocimiento del traductor humano con el sistema de traducción automática para lograr una traducción de alta calidad.

1.4. Traducción Automática Interactiva

Los sistemas de traducción modernos están lejanos de dar traducciones de calidad alta. Una alternativa dentro del amplio marco de trabajo de CAT, es el paradigma de la *Traducción Automática Interactiva* (a la que nos referiremos a lo largo de este trabajo como Interactive Machine Translation, utilizando sus siglas en inglés IMT), que puede ser vista como una evolución del escenario de post-edición de CAT.

El proyecto de investigación TransType [21] hizo una gran contribución a la tecnología IMT. En dicho proyecto la interacción humana se dirige hacia la traducción de texto y no como se venía haciendo en los anteriores proyectos interactivos a la desambiguación del texto a traducir. Con este proyecto por primera vez se incorporan técnicas de traducción automáticas basadas en datos dentro del entorno de traducción interactiva, teniendo como objetivo mejorar la eficiencia del sistema.

Más tarde, continuando la idea del TransType, [2] se planteó la utilización de un sistema de traducción automática que producía una hipótesis de traducción completa, permitiendo al usuario humano modificarla o aceptar la traducción. El sistema intenta hacer uso de la información que obtiene a partir de las acciones del usuario, para producir una nueva hipótesis y mejorar la calidad de las sugerencias. En cada interacción del proceso el usuario va a validar un prefijo como correcto y el sistema utilizará esta información para generar una nueva hipótesis que cumpla con el prefijo validado. El proceso concluye cuando el usuario da por válida toda la oración traducida. A este proceso se le denomina traducción automática interactiva.

En la Figura 1.2 podemos ver un ejemplo del paradigma IMT. El proceso se inicia cuando el usuario da una oración de entrada $\mathbf{x}$ para ser traducida. La referencia es la traducción que el usuario quiere lograr al final del proceso de IMT. En la iteración 0, el usuario no suministra ningún prefijo del texto al sistema, por este motivo $\mathbf{p}$ no contiene nada. Por tanto el sistema IMT tiene que proporcionar la traducción completa como sufijo $\mathbf{s}_h$, tal como si este fuera

un sistema convencional de SMT. En la siguiente iteración el usuario valida el prefijo $\mathbf{p}$ como correcto posicionando el cursor en la posición en la que comienza la primera palabra errónea, que en este caso se corresponde con el final del segmento "To". Por otra parte implícitamente se está marcando el resto de la oración, el sufijo $\mathbf{s}_l$ "switch on" como incorrecto. Posteriormente introduce una nueva palabra k que será la que él quiera a continuación del prefijo validado de modo que se asume que $k \neq \mathbf{s}_l$. En la próxima iteración el sistema brinda una hipótesis $\mathbf{s}_h$ y el usuario vuelve a validar el sufijo e introducir una nueva palabra. El proceso concluye cuando el usuario valide toda la oración y se introduzca el carácter especial $\#$. Aunque con IMT la obtención de los sufijos puede contener errores, el traductor humano tiene que hacer un menor número cambios que con un sistema SMT. Como puede verse en el ejemplo de la Figura 1.2, solo se tuvo que introducir una palabra en el lugar deseado para obtener la traducción de referencia, reduciendo el coste en tiempo y aumentado la productividad.

SOURCE $(\boldsymbol{x})$**:** Para encender la impresora:
REFERENCE $(\boldsymbol{y})$**:** To power on the printer:

ITER-0	$(\boldsymbol{p})$ $(\hat{\boldsymbol{s}}_h)$	( ) *To switch on:*
ITER-1	$(\boldsymbol{p})$ $(\boldsymbol{s}_l)$ (k) $(\boldsymbol{s}_h)$	To *switch on:* power *on the printer:*
ITER-2	$(\boldsymbol{p})$ $(\boldsymbol{s}_l)$ (k) $(\hat{\boldsymbol{s}}_h)$	To power on the printer: ( ) (#) ( )
FINAL	$(\boldsymbol{p} \equiv \boldsymbol{y})$	To power on the printer:

Figura 1.2: Ejemplo de funcionamiento del proceso IMT para traducir una oración del español al ingles. La "$\mathbf{p}$" indica los prefijos validados por el usuario, " $\mathbf{s}_h$" indica el sufijo propuesto por el sistema, " $\mathbf{s}_l$" sufijo incorrecto, "k" es la nueva palabra indicada por el usuario.

1.4.1. Traducción Automática Estadística Interactiva

La traducción automática estadística interactiva se basa en la formulación de SMT. Para esto es necesario modificar la Ecuación **1.1**, de acuerdo al escenario IMT teniendo en cuenta la parte de la oración que se encuentra traducida. Esta

formulación queda de la siguiente forma:

$$\hat{\mathbf{s}_h} = \underset{\mathbf{s}_h}{\operatorname{argmax}} Pr(\mathbf{s}_h|\mathbf{x}, \mathbf{p}, k) \tag{1.9}$$

donde el problema de maximización se define sobre el sufijo s_h, permitiéndonos reescribir la Ecuación 1.1 descomponiendo apropiadamente la parte derecha y eliminando los términos constantes, logrando el criterio equivalente

$$\hat{\mathbf{s}_h} = \underset{\mathbf{s}_h}{\operatorname{argmax}} Pr(\mathbf{p}, k, \mathbf{s}_h|\mathbf{x}) \tag{1.10}$$

La principal diferencia de la Ecuación 1.1 y 1.10 es en la búsqueda del argmax ya que en IMT se realiza sobre el sufijo $\mathbf{s}_h$ que complete $\mathbf{p}$ y k, en vez de la oración completa $\mathbf{y}$ como ocurre en SMT. Haciendo posible la utilización de los mismos modelos siempre y cuando el procedimiento de búsqueda se modifique correctamente [2].

1.4.2. Utilizando grafos de palabras

Un *grafo de palabras* es un grafo dirigido, acíclico y ponderado. Representa las posibles traducciones de una oración, donde cada nodo representa una hipótesis parcial de la traducción bien cada arista está etiquetada con una palabra de la oración destino, ponderada de acuerdo a la puntuación recibida por el modelo SMT [45].

Los grafos de palabras utilizados en IMT son una adaptación para la utilización en este paradigma de los segmentos de palabra o phrases. En este trabajo utilizaremos de manera semejante los grafos de palabras construidos, durante el procedimiento de búsqueda realizado en el modelo de SMT basado en segmentos. El modelo SMT basado en segmentos genera un grafo de segmentos, en lugar de un grafo de palabras, por lo que es necesario convertir este grafo de segmentos a grafo de palabras. Este procedimiento es bastante sencillo y consiste en añadir nodos y aristas entre cada una de las palabras que constituyen los segmentos y asignando la puntuación del segmento a la arista inicial. Para ver un ejemplo de este procedimiento lo podemos ver en la Figura 1.3. Para convertir el grafo de segmentos en grafo de palabras hay que tener en cuenta que las puntuaciones de las aristas no son probabilidades sino logaritmos de probabilidades ya que la maximización de la ecuación se realiza sin normalización. Estas puntuaciones dependen de dos factores: en primer lugar de la función de características asociada a la oración representada en el grafo y en segundo lugar a los pesos del modelo log-lineal asociados a esta función, por lo que depende de ambos conjuntos de parámetros para obtener la puntuación de transición entre los nodos del grafo. En este trabajo final de máster se pretende optimizar únicamente el valor de los pesos del modelo log-lineal para

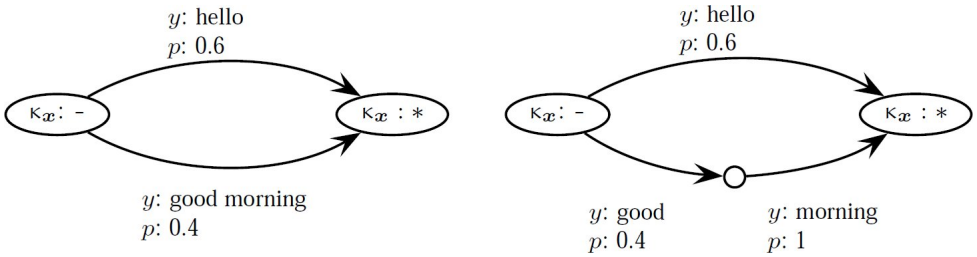


Figura 1.3: Ejemplo de procedimiento para convertir un grafo de segmentos (izquierda) a grafo de palabras (derecha) para IMT Los símbolos K_x es el vector de la cobertura de la frase de entrada, el símbolo $-$ indica una palabra no cubierta, y el símbolo $*$ una palabra de entrada que ya ha sido traducida. En cada arco se presenta la palabra emitida cuando transita por ese camino, y la probabilidad que tiene asignada.

una oración $\mathbf{x}$ dada. Por lo tanto, asumimos que el grafo de palabras dependerá solamente del conjunto de pesos $\boldsymbol{\lambda}$ y no de las funciones características, denotándolo como $W(\mathbf{x})$.

Cuando se realiza un proceso IMT para una oración dada, el sistema hace uso del grafo de palabras generado para la oración. Con él se trata de completar el prefijo aceptado por el traductor humano. El sistema busca específicamente encontrar el mejor camino en el grafo de palabras asociado con el prefijo dado, de forma que permita completar la traducción, siendo capaz de proporcionar muchas sugerencias de terminación para cada prefijo.

Los grafos de palabras contienen solamente un subconjunto de los todas la posibles secuencias de palabras. ¿Qué ocurre cuando el usuario introduce un prefijo que no se encuentra en el grafo de palabras? El sistema no puede encontrar dentro del grafo un sufijo adecuado. Para solucionar este problema se realiza una búsqueda tolerante en el grafo de palabras[30], empleando un modelo corrector de errores. Se selecciona el conjunto de nodos que correspondan a la secuencia de palabras con la menor distancia al prefijo dado, calculando esta distancia mediante el algoritmo de Levenshtein. Una vez obtenido ese conjunto de nodos, se elige el que tenga la máxima probabilidad y se calcula el camino a partir de este. Gracias a este método la hipótesis que se muestra al usuario contiene las palabras que son parte del prefijo.

1.5. Adaptación del sistema

La calidad de las traducciones disminuye cuando los textos a traducir pertenecen a dominios diferentes a los corpus de entrenamiento o desarrollo utilizado en el sistema [7] . Esto hace que la *adaptación* sea un problema muy común dentro de la traducción automática estadística. La adaptación tiene como objetivo mejorar el rendimiento de los sistemas entrenados y calibrados mediante

corpus de entrenamiento y desarrollo fuera del dominio del texto a traducir.

Existen varias técnicas que realizan una adaptación eficaz como el *filtrado* [24], en donde se exploran las oraciones de entrenamiento para buscar datos similares al dominio de interés. Las oraciones encontradas pueden emplearse para aumentar el tamaño del conjunto de desarrollo o bien para crear nuevos modelos con los nuevos datos dentro del dominio y finalmente interpolarlos con los datos fuera del dominio. También se puede realizar un nuevo muestreo en las oraciones del conjunto de entrenamiento para ponderar su relevancia en la tarea del dominio específico [38] [39] [14]. A pesar de que estas técnicas ofrecen gran interés el sistema no se adapta al procesar cada una de sus oraciones, sino al procesar la totalidad de las oraciones.

Para realizar adaptación de forma online, se pueden ajustar las funciones de características o los pesos log-lineales con el objetivo de mejorar el rendimiento de los sistemas modificando sus estimaciones de probabilidad. Por ello, en este trabajo final de máster seguiremos la estrategia de adaptar los pesos del modelo log-lineal, con el objetivo de mejorar el rendimiento del sistema.

En CAT e IMT las tareas son cambiantes por lo que el problema de adaptación es más interesante. Las estrategias de adaptación anteriormente mencionadas no se pueden aplicar de forma directa en los marcos CAT e IMT. El proceso de adaptación se realiza utilizando un conjunto de frases traducidas que son producidas por el usuario a partir de oraciones de entrada para un idioma de origen dado.

Las estrategias de adaptación en traducción automática tradicional pueden requerir de cantidad de recursos computacionales y tiempo. Este proceso no conlleva ningún problema porque se realiza antes de comenzar la tarea de traducción. En los paradigmas CAT e IMT esto es muy diferente, en este caso los sistemas trabajan bajo la restricción de la interacción del usuario y los textos a traducir pueden variar constantemente. Con estos problemas, no es conveniente utilizar la adaptación con la totalidad de las oraciones, ya que el coste de realizar la optimización es muy alto después de traducir cierto número de oraciones. La idea es adaptar los modelos de forma online, haciendo uso de las correcciones proporcionadas por el usuario, sin tener que reentrenar completamente los parámetros del modelo, ahorrando de esta forma un gran coste. Se han propuesto diferentes aproximaciones para abordar el problema de la adaptación online basándose en traducción oración tras oración. En [32] los autores proponen un sistema de aprendizaje online para IMT en donde los modelos que intervienen en el proceso de traducción se actualizan de forma incremental mediante una versión incremental del algoritmo expectation-maximisation (EM) [12], permitiendo la inclusión de nuevos pares de frases en el sistema.

En este trabajo final de máster uno de los propósitos es adaptar los mecanismos de predicción a IMT empleando varias estrategias de adaptación. Este

tipo de adaptación permite comparar diferentes estrategias de aprendizaje online para adaptar $\boldsymbol{\lambda}$. Para abordar la tarea de adaptación online en un escenario IMT, todo este trabajo se basará en [16], donde se plantean diferentes estrategias para adaptar los pesos del modelo log-lineal empleando un sistema CAT no interactivo y en [37] donde se plantea una estrategia para adaptar los pesos del modelo log-lineal empleando un sistema IMT.

1.6. Conclusiones

Existe gran necesidad de comunicación entre la personas y el idioma puede resultar un obstáculo por lo que el uso del ordenador para la traducción automática ha sido objeto de estudio desde mediados del siglo XX. La traducción automática estadística ha aportado los mejores resultados, siendo la más utilizada tanto de forma empresarial como universitaria. A pesar de los resultados obtenidos en la traducción automática estadística es necesario la intervención de los traductores humanos para lograr la calidad de traducción deseada, dando lugar a sistemas de traducción automática interactivos que permiten aumentar la calidad de la traducción con el menor esfuerzo posible por parte del traductor humano. La diversidad de los textos a introducir es muy grande por lo que la adaptación es un tema de interés dentro de la traducción automática. El problema de adaptación es de gran atención cuando el sistema debe ser continuamente adaptado por ejemplo con cada interacción humana. En este trabajo final de máster se pretende llevar a cabo varias aproximaciones para adaptar los pesos del modelo log-lineal dentro del marco de la traducción interactiva.

Capítulo 2

Aprendizaje Online de los pesos

En este trabajo se pretende adaptar el paradigma de aprendizaje online donde se adaptan los pesos del modelo log-lineal para mejorar la calidad de la traducción en el marco IMT. En el paradigma de aprendizaje online, las etapas de entrenamiento y predicción ya no están separadas. Esto es muy útil en IMT ya que de esta forma se puede lograr que el sistema vaya aprendiendo por la realimentación que se proporciona por cada interacción con el usuario. Los algoritmos tradicionales de ajuste pretenden encontrar el conjunto de pesos del modelo log-lineal que maximice la calidad de las traducciones. Algunos de estos algoritmos procesan todas las muestras del conjunto de desarrollo tantas veces como sea necesaria para lograr la convergencia del algoritmo. Sin embargo, en un sistema IMT o CAT, para llevar a cabo la adaptación del sistema, es temporal y computacionalmente ineficiente procesar todas las muestras cada vez que vaya apareciendo una nueva. Debido a lo planteado, después de procesar una muestra, el sistema realiza la siguiente predicción realimentándose con información que el usuario proporciona de la última muestra procesada pero sin procesar todas las anteriores. La información que proporciona la realimentación puede ir desde una simple opinión sobre la calidad de la predicción que ha realizado el sistema, como por ejemplo aceptando parte de la hipótesis como ocurre en IMT, o dando la etiqueta real de la muestra procesada en entornos completamente supervisados. La Figura 2.1, nos muestra el paradigma de *aprendizaje online* dentro de IMT, incorporando la realimentación del usuario dentro del *módulo estimador* $\boldsymbol{\lambda}$ y la interacción entre el usuario y sistema IMT en cada interacción k . El objetivo del *módulo estimador* $\boldsymbol{\lambda}$ es generar el conjunto de pesos que se utilizará en la creación del grafo de palabras de la siguiente frase. Cuando el sistema SMT recibe una nueva frase de entrada $\mathbf{x}$ en un idioma origen, el sistema SMT genera un grafo de palabras a partir de la frase de entrada. Seguidamente el sistema IMT va a recibir el grafo de palabras para generar

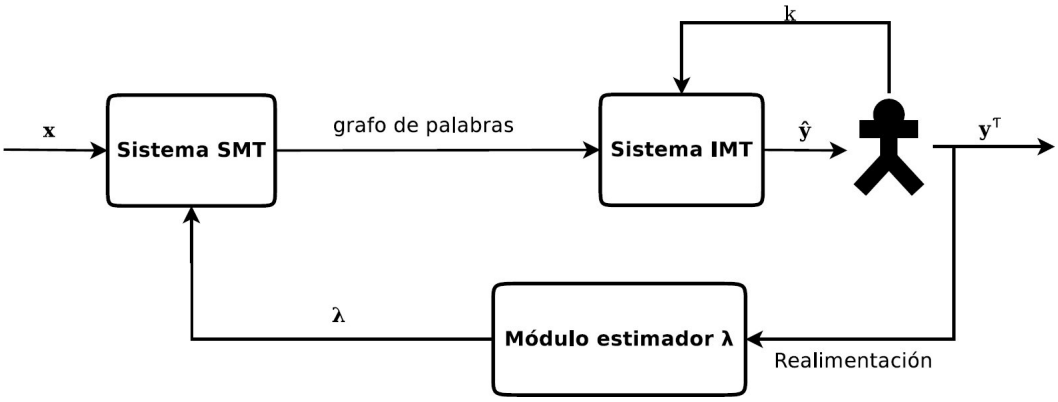


Figura 2.1: Esquema del paradigma de aprendizaje online dentro del marco de la IMT

a partir de él el mejor sufijo, mostrando al usuario el mejor camino del grafo de palabras. En cada interacción entre el sistema y el traductor humano este puede aceptar un prefijo o la traducción en su totalidad, concluyendo el proceso cuando el traductor humano acepta en su totalidad la traducción propuesta por el sistema, convirtiéndose esta en la traducción $\mathbf{y}^{\tau}$. Esta oración se utilizará como realimentación para que el algoritmo de aprendizaje online modifique el valor de $\boldsymbol{\lambda}$ con el objetivo de mejorar la calidad de las futuras traducciones. A partir de la segunda oración que entra en al sistema SMT, se modifican el conjunto de pesos que se utiliza en la generación del grafo por el conjunto de pesos que se optimizó en el *Módulo estimador* $\boldsymbol{\lambda}$. Como consecuencia de esta modificación, el peso de las diferentes aristas del grafo de palabras también será modificado.

La ecuación del *modelo log-lineal* en cada iteración queda definida de la siguiente manera:

$$\hat{\mathbf{y}} = \underset{\mathbf{y}}{\text{argmax}} \sum_{m=1}^{M} \lambda_m^t h_m(\mathbf{x}_t, \mathbf{y}) \tag{2.1}$$

definida de forma vectorial como:

$$\hat{\mathbf{y}} = \underset{\mathbf{y}}{\text{argmax}}\, \boldsymbol{\lambda}_t \mathbf{h}(\mathbf{x}_t, \mathbf{y}) \tag{2.2}$$

en donde los pesos del modelo log-lineal $\boldsymbol{\lambda}_t$ varían de acuerdo a la muestra $(\mathbf{x}_{t-1}, \mathbf{y}_{t-1}^{\tau})$ vista antes del momento t.

El objetivo es obtener traducciones con la mayor calidad posible, que se asemejen lo máximo posible a la traducciones de referencia $\mathbf{y}^{\tau}$. Aunque es frecuente que la hipótesis $\hat{\mathbf{y}}$ que maximiza la verosimilitud no es la hipótesis de mayor calidad $\mathbf{y}^{*}$, según la perspectiva de un traductor humano o una medida de calidad dada. Por tanto, es posible que la hipótesis con mayor calidad $\mathbf{y}^{*}$ no coincida con la hipótesis con mayor verosimilitud $\hat{\mathbf{y}}$.

Otro aspecto a tener en cuenta es que $\mathbf{y}^*$ puede no coincidir con $\mathbf{y}^\tau$, la oración de referencia, debido a eventuales problemas de cobertura. Basándonos en lo realizado en [16], se propone adaptar los parámetros del modelo de forma que $\mathbf{y}^*$ obtenga la mayor puntuación de acuerdo a la ecuación 1.7.

Se define la diferencia entre la calidad de la hipótesis propuesta por el sistema $\hat{\mathbf{y}}$ y la mejor hipótesis $\mathbf{y}^*$ en función de la medida de calidad $\mu(\cdot)$

$$l(\hat{y}) = |\mu(\hat{y}) - \mu(y^*)| \tag{2.3}$$

Para evaluar un sistema SMT se pueden emplear diferentes medidas de calidad. Por ejemplo, TER [41] representa una tasa de error, mientras que BLEU [33] representa una medida de precisión. Por ello se incluyó el valor absoluto en la ecuación 2.3 para preservar la generalidad. Además, la diferencia entre los valores de $\hat{\mathbf{y}}$ e $\mathbf{y}^*$ ha sido definida como:

$$\phi(\hat{\mathbf{y}}) = s(\mathbf{x}, \mathbf{y}^*) - s(\mathbf{x}, \hat{\mathbf{y}}) \tag{2.4}$$

Nos gustaría que $l(\cdot)$ y $\phi(\cdot)$ estén relacionadas de forma que las diferencias de una se correspondan con las de la otra. Una hipótesis candidata $\mathbf{y}$ con calidad de traducción $\mu(\mathbf{y})$, debería ser muy similar a la calidad de la traducción proporcionada por $\mu(\mathbf{y}^*)$, esperando que $s(\mathbf{x}; \mathbf{y})$ sea muy similar a $s(\mathbf{x}; \mathbf{y}^*)$. Por lo tanto, el propósito de nuestro procedimiento online debe ser promover esta correspondencia después de procesar la muestra t.

En el trabajo [16] se proponen dos aproximaciones, adaptar en el instante de tiempo t la función de características $\mathbf{h}_t$ o los pesos del modelo log-lineal $\boldsymbol{\lambda}_t$ para SMT. En este trabajo final de máster únicamente nos centraremos en la adaptación de los pesos del modelo log-lineal para un escenario IMT. Esta aproximación ofrece mejores resultados probablemente debido a que únicamente deben adaptarse el mismo número de parámetros que modelos intervienen en la combinación log-lineal, alrededor de 14 parámetros, frente a los millones en el caso de adaptar $\mathbf{h}_t$.

2.1. Adaptación de los factores de escalado

En este trabajo vamos a emplear varias técnicas para adaptar los factores de escalado del modelo log-lineal $\boldsymbol{\lambda}$, para abordar el problema del aprendizaje online en IMT. El objetivo es ajustar el poder discriminatorio de cada modelo de forma que la puntación resultante sea mayor para la hipótesis que es más parecida a la oración de referencia $\mathbf{y}^\tau$ que la puntuación de cualquier otra hipótesis.

Cuando el sistema recibe una oración de entrada $\mathbf{x}_t$ y su correspondiente oración de referencia $\mathbf{y}_t^\tau$, hay que obtener el termino de actualización $\check{\boldsymbol{\lambda}}_t$ para la oración observada en el instante t. Una vez calculado $\check{\boldsymbol{\lambda}}_t$ se utilizará para

actualizar el valor de $\boldsymbol{\lambda}$ de la siguiente forma:

$$\boldsymbol{\lambda_t} = (1-\alpha)\boldsymbol{\lambda_{t-1}} + \alpha\tilde{\boldsymbol{\lambda}}_{\boldsymbol{t}} \tag{2.5}$$

empleando un ratio de aprendizaje α. Hay que tener en cuenta que para calcular $\boldsymbol{\lambda_t}$ primeramente hay que haber calculado $\tilde{\boldsymbol{\lambda}}_{\boldsymbol{t}}$ para un par de frases en un instante $t-1$.

2.2. Métodos

Cuando se desea traducir una oración $\mathbf{x}$ en un idioma origen utilizando un sistema IMT, este generará un grafo de palabras que contendrá todas las posibles traducciones para la oración de entrada $\mathbf{x}$ en un idioma destino. El sistema escogerá entre todas las hipótesis contenidas en el grafo de palabras la mejor hipótesis, la cual será aquella cuyo camino contenga la mayor puntuación. Para esto se emplearan diversos algoritmos de aprendizaje online para ajustar los pesos del modelo log-lineal de manera online.

En las siguientes secciones se plantearan algunos de los algoritmos de aprendizaje online empleados en este trabajo. En primer lugar, los explicaremos para un escenario post-edición y luego la adaptación para IMT. Cabe destacar que, por simplicidad, siguiendo con la nomenclatura presentada en[23], se omitirán los subíndices y superíndices t en caso de que no se requiera de una clara distinción temporal.

2.2.1. Discriminative ridge regression

El algoritmo discriminative ridge regression[23] (al que nos referiremos a lo largo de este trabajo como DRR, utilizando sus siglas en ingles) es un algoritmo de adaptación online discriminativo. En este algoritmo se tiene en cuenta todas las hipótesis dentro de la lista N-best intentando encontrar el mejor conjunto de pesos tal que las buenas hipótesis dentro de la lista N-best tengan la puntuación más alta y penalizando a las peores con una puntuación baja.

En post edición el algoritmo utiliza toda la lista N-best en orden decreciente, al contrario de otros algoritmos que solamente utilizan la mejor traducción.
Pasos para adaptar $\boldsymbol{\lambda}$ mediante el algoritmo DRR:

- Definir la matriz $\mathbf{H_x}$, de talla *N x M* donde *M* es el número de características que contienen las funciones de características $\mathbf{h}$ de cada frase y *N* el número de frases dentro de la lista N-best.

$$\mathbf{H_x} = [\mathbf{h}(\mathbf{x}, \mathbf{y}_1), ..., \mathbf{h}(\mathbf{x}, \mathbf{y}_N)]' \tag{2.6}$$

- A continuación definimos $\mathbf{H}^*_\mathbf{x}$, como una matriz donde sus filas van a ser iguales al vector de características de la mejor hipótesis $\mathbf{y}^*$ de la lista N-best.

$$\mathbf{H}^*_\mathbf{x} = [\mathbf{h}(\mathbf{x}, \mathbf{y}^*), ..., \mathbf{h}(\mathbf{x}, \mathbf{y}^*)]' \tag{2.7}$$

- $\mathbf{R}_\mathbf{x}$ se define de la siguiente manera:

$$\mathbf{R}_\mathbf{x} = \mathbf{H}^*_\mathbf{x} - \mathbf{H}_\mathbf{x} \tag{2.8}$$

- El algoritmo DRR pretende encontrar el vector $\check{\boldsymbol{\lambda}}$ que relacione las diferencias en valores como la diferencia en la calidad de las diferentes hipótesis.

$$\mathbf{R}_\mathbf{x} \cdot \check{\boldsymbol{\lambda}}_t \propto \mathbf{l}_\mathbf{x} \tag{2.9}$$

  siendo $\mathbf{l}_\mathbf{x}$ un vector columna de N filas representado como:

$$\mathbf{l}_\mathbf{x} = [l(\mathbf{y}_1), ..., l(\mathbf{y}), ..., l(\mathbf{y}_N)]', \forall \mathbf{y} \in nbest(\mathbf{x}) \tag{2.10}$$

  Por lo tanto se define $\check{\boldsymbol{\lambda}}_t$ a buscar como:

$$\check{\boldsymbol{\lambda}}_t = \underset{\boldsymbol{\lambda}}{\text{argmin}} \, |\mathbf{R}_\mathbf{x} \cdot \boldsymbol{\lambda} - \mathbf{l}_\mathbf{x}| \tag{2.11}$$

$$= \underset{\boldsymbol{\lambda}}{\text{argmin}} \, ||\mathbf{R}_\mathbf{x} \cdot \boldsymbol{\lambda} - \mathbf{l}_x||^2 \tag{2.12}$$

- A pesar de que las ecuaciones 2.11 y 2.12 son equivalentes, gracias a la regresión de arista $\check{\boldsymbol{\lambda}}_t$ puede resolverse como la solución al sistema sobre determinado $\mathbf{R}_\mathbf{x} \cdot \check{\boldsymbol{\lambda}}_t = \mathbf{l}_\mathbf{x}$ de la siguiente forma:

$$\check{\boldsymbol{\lambda}}_t = (\mathbf{R}'_\mathbf{x} \cdot \mathbf{R}_\mathbf{x} + \beta\mathbf{I})^{-1} \cdot \mathbf{l}_\mathbf{x} \tag{2.13}$$

  donde β es un valor pequeño que representa el termino de regularización para estabilizar el producto $\mathbf{R}'_x \, \mathbf{R}_x$ y asegurar que este sea invertible.

Discriminative ridge regression en escenario interactivo

Al aplicar el algoritmo DRR en un escenario IMT, la métrica de calidad que empleamos no está relacionada a una sola hipótesis sino a todo un grafo de palabras. En un sistema IMT es común evaluar la calidad calculando el número de interacciones que un usuario realiza para modificar la hipótesis hasta obtener la oración deseada. Para medir esto se utiliza el KSMR [2], esta métrica es empleada para medir la calidad de un sistema IMT.

El número de interacciones no se puede calcular en función de la hipótesis $\mathbf{y}$, puesto que cada vez que se introduce una palabra el sistema IMT modifica el

sufijo y por lo tanto se debe calcular simulando el procedimiento de interacción con la ayuda de un grafo de palabras.

La idea es que al optimizar la métrica de calidad se disminuya el número de interacciones con el usuario para obtener la oración de referencia. Para ello se aplicó el algoritmo DDR en un escenario de post-edición, demostrando que esta idea no es completamente adecuada como se verá en más adelante. Por lo tanto, la métrica que se optimiza mediante el aprendizaje online no va a depender únicamente de la mejor hipótesis, por lo que es necesario modificar la formulación del algoritmo DRR descrito en la primera parte de está sección.

Primeramente la lista N-best se puede considerar una lista N-best grafos de palabras, aunque este concepto es confuso porque no hay forma de saber cuales son las N-best en IMT. Por lo que en vez de calcular una verdadera lista N-best grafos de palabras calcularemos el grafo de palabras $W_{\boldsymbol{\lambda}^n}(\mathbf{x})$ asociado a cada frase de entrada $\mathbf{x}$ a partir de un conjunto de pesos $\boldsymbol{\lambda}^n$, entre los obtenidos previamente mediante diferentes aproximaciones que se explicaran en secciones posteriores $\boldsymbol{\Lambda} = \{\boldsymbol{\lambda}^1, \ldots, \boldsymbol{\lambda}^n, \ldots, \boldsymbol{\lambda}^N\}$. Estos grafos de palabras que son generados a partir de los pesos obtenidos mediante diferentes aproximaciones, no constituyen una verdadera lista de los N-best grafos de palabras, sino una aproximación de esta. Asumimos que el mejor camino del grafo es su representación, aunque esto puede no ser lo idoneo.

En este caso el algoritmo DRR va a premiar a los grafos de palabras que tengan alta calidad y va a penalizar aquellos grafos de palabras con baja calidad. Lo realmente importante es tener grafos de palabras $W_{\boldsymbol{\lambda}^n}(\mathbf{x})$ con diferente calidad, realmente lo que estamos premiando y penalizando es a un conjunto $\boldsymbol{\lambda}^n$. De esta forma proponemos el vector columna $\mathbf{l_x}$ cuyas N filas son:

$$\mathbf{l_x} = [l(W_{\boldsymbol{\lambda}^1}(\mathbf{x})) \ldots l(W_{\boldsymbol{\lambda}^n}(\mathbf{x})) \ldots l(W_{\boldsymbol{\lambda}^N}(\mathbf{x}))] \tag{2.14}$$

La matriz $\mathbf{H_x}$ es otro de los aspectos importantes en el algoritmo. En el marco IMT debe ser redefinida ya que el conjunto de características no son de las hipótesis del conjunto N-best, sino que son de los grafos de palabras obtenidos del conjunto de pesos. El grafo de palabras tiene un vector de características para cada uno de los caminos, por lo que el vector de características que se empleará será el del mejor camino de $W_{\boldsymbol{\lambda}^n}(\mathbf{x})$, es decir, el vector de características de la mejor hipótesis de $W_{\boldsymbol{\lambda}^n}(\mathbf{x})$.
Definiendo $\mathbf{H_x}$ para IMT de la siguiente manera:

$$\mathbf{H_x} = [\mathbf{h}_{\boldsymbol{\lambda}^1}, ..., \mathbf{h}_{\boldsymbol{\lambda}^n}]' \tag{2.15}$$

Se define $\mathbf{H}^*$,

$$\mathbf{H}^*_{\mathbf{x}} = [\mathbf{h}_{\boldsymbol{\lambda}^*}, ..., \mathbf{h}_{\boldsymbol{\lambda}^*}]' \tag{2.16}$$

donde $\mathbf{h}_{\boldsymbol{\lambda}^*}$ es el vector de características de la mejor hipótesis del grafo de palabras $W_{\boldsymbol{\lambda}^*}(\mathbf{x})$ y $W_{\boldsymbol{\lambda}^*}(\mathbf{x})$ es el grafo de mayor calidad dentro de los diferentes

$\boldsymbol{\lambda}$ del conjunto $\boldsymbol{\Lambda}$. Para medir la calidad se utiliza la métrica KSMR empleada en IMT.

2.2.2. Passive Aggressive

El algoritmo Passive Aggressive [11] [23] (al que nos referiremos a lo largo de este trabajo como PA, utilizando sus siglas en inglés). En este caso se aplica a un problema de regresión, donde se tiene que obtener la frase de salida $\mathbf{y}$ para la frase de entrada en el instante de tiempo *t*. Para lograr esto se utiliza una *regresión lineal* para aprender el conjunto de pesos. Si el error cometido es menor a un parámetro de sensibilidad, la pérdida sufrida por el sistema es 0 y el algoritmo permanece pasivo, es decir, el nuevo vector de pesos es igual al anterior. En caso contrario, la pérdida crece linealmente con respecto al error y el algoritmo pasa a modo agresivo actualizando los parámetros. La idea detrás del algoritmo PA es calcular los pesos de la función de regresión, de modo que la función de pérdida sea lo más cercana a 0, con el objetivo de ser posible que el vector de pesos sea lo más parecido al anterior.

En [23] se propone se adaptar los factores de escalado $\boldsymbol{\lambda}$ utilizando el algoritmo PA.

- $\check{\boldsymbol{\lambda}}_t$ puede calcularse:

$$\check{\boldsymbol{\lambda}}_t = \boldsymbol{\phi}(\hat{\mathbf{y}}) \frac{\sqrt{l(\hat{\mathbf{y}})} - \boldsymbol{\lambda}_{t-1}\boldsymbol{\phi}(\hat{\mathbf{y}})}{||\boldsymbol{\phi}(\hat{\mathbf{y}})||^2 + \frac{1}{C}} \tag{2.17}$$

  donde C es denominado factor de agresividad, $l(\hat{\mathbf{y}})$ es la diferencia en la calidad de la traducción entre la hipótesis propuesta por el sistema y la mejor hipótesis como se describe en Ecuación 2.3

- $\boldsymbol{\phi}(\hat{\mathbf{y}})$ es la diferencia del conjunto de características de la mejor traducción $\mathbf{y}^*$ y la traducción de la frase $\hat{\mathbf{y}}$:

$$\boldsymbol{\phi}(\hat{\mathbf{y}}) = \mathbf{h}(\mathbf{x}, \mathbf{y}^*) - \mathbf{h}(\mathbf{x}, \hat{\mathbf{y}}) \tag{2.18}$$

- se va a realizar la actualización cuando no se cumpla lo siguiente:

$$\boldsymbol{\lambda}_{t-1}\boldsymbol{\phi}(\hat{\mathbf{y}}) \geq \sqrt{l(\hat{\mathbf{y}})} \tag{2.19}$$

Passive Aggressive en escenario interactivo

Al igual que lo descrito en la sección 2.2.1, el número de interacciones entre el usuario y el sistema no se puede calcular en función de la hipótesis $\mathbf{y}$, porque se puede modificar el sufijo cada vez que se introduce una palabra. Se debe simular el procedimiento de interacción con la ayuda de un grafo de palabras. El algoritmo PA descrito en la sección anterior en escenario post-edición no

se puede emplear directamente en un escenario IMT, por lo que es necesario modificar la formulación del algoritmo.

El algoritmo PA no tiene en cuenta todas las hipótesis dentro la lista N-best como es el caso de DRR. Solamente se utiliza la mejor hipótesis de la traducción $\mathbf{y}^*$ y la traducción de la frase propuesta por el sistema $\hat{\mathbf{y}}_t$. Para seleccionar la mejor hipótesis se utiliza el grafo de palabras $W_{\boldsymbol{\lambda}^*}(\mathbf{x})$ con el mejor camino dentro de los diferentes $\boldsymbol{\lambda}$ del conjunto $\boldsymbol{\Lambda}$. Para medir esta calidad utilizaremos la métrica KSMR, por lo que $\mathbf{h}(\mathbf{x}, \mathbf{y}^*)$ es el vector de características del mejor camino del grafo de palabras $W_{\boldsymbol{\lambda}^*}(\mathbf{x})$ de mayor calidad. Para obtener la frase propuesta por el sistema $\hat{\mathbf{y}}_t$ se va a utilizar el grafo de palabras $W_{\hat{\boldsymbol{\lambda}}}(\mathbf{x})$ obtenido a partir del conjunto de pesos baseline $\hat{\boldsymbol{\lambda}}$. El conjunto de características seleccionadas $\mathbf{h}(\mathbf{x}, \hat{\mathbf{y}})$ son las características del mejor camino asociado al grafo de palabras $W_{\hat{\boldsymbol{\lambda}}}(\mathbf{x})$.

$\boldsymbol{\phi}(\hat{\mathbf{y}})$ es la diferencia entre el conjunto de características $\mathbf{h}_{\boldsymbol{\lambda}^*}$ del mejor camino del grafo de palabras de mayor calidad $W_{\boldsymbol{\lambda}^*}(\mathbf{x})$ y el conjunto de características $\mathbf{h}_{\hat{\boldsymbol{\lambda}}}$ del mejor camino del grafo de palabras obtenido a partir del conjunto de pesos baseline $W_{\hat{\boldsymbol{\lambda}}}(\mathbf{x})$:

$$\boldsymbol{\phi}(\hat{\mathbf{y}}) = \mathbf{h}_{\boldsymbol{\lambda}^*} - \mathbf{h}_{\hat{\boldsymbol{\lambda}}} \tag{2.20}$$

Para calcular $l(\hat{\mathbf{y}})$ que representa la calidad de la hipótesis queda redefinido de la siguiente forma:

$$l(\hat{\mathbf{y}}) = l(W_{\hat{\boldsymbol{\lambda}}}(\mathbf{x})) \tag{2.21}$$

donde $l(W_{\hat{\boldsymbol{\lambda}}}(\mathbf{x}))$ representa la diferencia entre la calidad del mejor camino del grafo de palabras obtenido a partir del conjunto de pesos baseline $W_{\hat{\boldsymbol{\lambda}}}(\mathbf{x})$ con respecto al mejor camino del grafo de palabras de mayor calidad.

2.2.3. Perceptron

Aunque el algoritmo Perceptron [36][10] es muy popular, en este trabajo se utilizará una variación de él, el Perceptron-Like [13], algoritmo también muy utilizado, el cual brinda un soporte sencillo y gran velocidad. El exito de aplicación del Percetron-Like se puede ver en [13] [16]. El algoritmo de Perceptron-Like es un algoritmo de error que estima los pesos de una combinación lineal de funciones mediante la comparación de la salida del sistema $\hat{\mathbf{y}}$ con respecto a la verdadera etiqueta $\mathbf{y}^*$ de la correspondiente entrada $\mathbf{x}$. El algoritmo Perceptron-Like permanece ejecutándose un cierto número de veces por el conjunto de muestras hasta alcanzar la convergencia. En nuestro caso de adaptación online para el marco de IMT, el algoritmo Perceptron-Like no visitará una muestra de nuevo después de ser procesada.

Para adaptar los factores de escalado es necesario disponer de los valores del conjunto de características de la traducción propuesta por el sistema

$\mathbf{h}(\mathbf{x}_t, \hat{\mathbf{y}}_t)$ y los valores del vector de características de la mejor traducción de la frase $\mathbf{h}(\mathbf{x}, \mathbf{y}^*)$, que van a ser combinados linealmente con el factor de escalado $\boldsymbol{\lambda}$ y el término de actualización $\check{\boldsymbol{\lambda}}_t$ el cual se calcula como:

$$\check{\boldsymbol{\lambda}}_t = sign(\mathbf{h}(\mathbf{x}, \mathbf{y}^*) - \mathbf{h}(\mathbf{x}, \hat{\mathbf{y}})) \tag{2.22}$$

El operador $sign$ indica que se tomarán medidas de tamaño fijo en la adaptación de los parámetros es decir $(+1, -1)$.

Perceptron en escenario interactivo

Como se ha visto en los algoritmos anteriormente planteados, este no se puede emplear directamente en un escenario IMT. Por lo tanto el algoritmo Perceptron-Like planteado en escenario post edición ha de ser modificado para el marco IMT.

En este caso se va a considerar, como en la sección 2.2.2, el vector de características de la mejor hipótesis $\mathbf{h}(\mathbf{x}, \mathbf{y}^*)$, como el vector de características del mejor camino del grafo de palabras $W_{\boldsymbol{\lambda}^*}(\mathbf{x})$ de mayor calidad obtenido a partir de los pesos del conjunto $\boldsymbol{\Lambda}$ y el conjunto de características de la hipótesis propuesta por el sistema $\mathbf{h}(\mathbf{x}, \hat{\mathbf{y}})$ será el conjunto de características asociadas al mejor camino del grafo de palabras obtenido $\hat{\boldsymbol{\lambda}}$. Redefiniendo la ecuación 2.22 del término de actualización $\check{\boldsymbol{\lambda}}_t$ de la siguiente manera:

$$\check{\boldsymbol{\lambda}}_t = sign(\mathbf{h}_{\boldsymbol{\lambda}^*} - \mathbf{h}_{\hat{\boldsymbol{\lambda}}}) \tag{2.23}$$

2.3. Generación del conjunto de pesos

El conjunto de pesos empleados para la generación del grafo de palabras es de gran influencia para la calidad del mejor camino, influyendo en la calidad de la traducción de la oración a partir de la cual se creó el grafo de palabras. Por tanto, si se emplean conjuntos de pesos obtenidos de forma aleatoria la calidad de los grafos de palabras seria muy baja. Para evitar esto vamos a generar un conjunto finito de nuevos conjuntos de pesos $\boldsymbol{\Lambda}$, partiendo del conjunto de pesos baseline $\hat{\boldsymbol{\lambda}}$, calculado por la técnica del MERT. Estos nuevos conjuntos van a ser similares al obtenido con el MERT. De esta manera pretendemos obtener grafos de palabras con una calidad superior a los obtenidos por el sistema de referencia basándonos en que el conjunto de pesos generados por el MERT no es el mejor posible para el dominio. A continuación planteamos las dos aproximaciones que utilizamos para obtener $\boldsymbol{\Lambda}$.

2.3.1. Aproximación gaussiana

La primera aproximación que utilizamos es muy similar a la utilizada en [37]. Se utiliza una *distribución gaussiana*. Basándose en la probabilidad que el con-

junto de pesos generado mediante MERT no sea el mejor posible para el dominio de las oraciones a traducir, debido a que el conjunto de desarrollo empleado para ajustar los pesos mediante MERT pertenece a un dominio diferente.

El método para generar cada conjunto de pesos perteneciente a $\mathbf{\Lambda}$ de forma semi-aleatoria es realizado mediante una distribución gaussiana. Donde las medias se corresponden con $\hat{\boldsymbol{\lambda}}$, es decir los pesos obtenidos con MERT. De esta manera semi-aleatoria se espera que los conjuntos de pesos que se utilizarán para la generación de los grafos tengan una variedad suficiente pero que no sean completamente aleatorios.

2.3.2. Aproximación simplex

El segundo método utilizado para generar los conjuntos de pesos está basado en el algoritmo de programación lineal *Simplex*[27], algoritmo iterativo que tiene como objetivo buscar el mínimo o máximo de una función.

Lo que buscamos es disminuir la cantidad de interacciones que el usuario debe realizar con el sistema para lograr la traducción deseada. Para medir esto utilizamos la métrica KSMR, y por tanto la métrica a optimizar por el simplex, siendo un problema de minimización.

Para el algoritmo simplex es necesario tener un conjunto de variables iniciales, en este caso son los pesos $\hat{\boldsymbol{\lambda}}$ obtenidos mediante el MERT. El aspecto más importante dentro del algoritmo simplex es la definición de la función objetivo. En nuestro caso la función objetivo va a ser el proceso IMT.

En cada iteración el algoritmo modifica el conjunto de variables iniciales, con el objetivo de minimizar la función objetivo, es decir, el valor de KSMR. Al concluir el algoritmo tenemos el conjunto $\mathbf{\Lambda} = \{\boldsymbol{\lambda}^1, \ldots, \boldsymbol{\lambda}^n, \ldots, \boldsymbol{\lambda}^N\}$ donde N es igual al número de iteraciones del algoritmo y $\boldsymbol{\lambda_n}$ es el conjunto de pesos que se utilizó en la iteración n, para la generación del grafo de palabras de la frase que se utilizará en el proceso IMT.

2.4. Conclusiones

La adaptación de los pesos del modelo log-lineal es una de las aproximaciones que existen para llevar a cabo la adaptación online. En este trabajo final de máster se ha decidido escogerla por ser una de las que tiene mejor rendimiento.

Existen muchos algoritmos de aprendizaje online, el algoritmo Discriminative Ridge Regression es uno de los que ha obtenido los mejores resultado en un problema de adaptación. En este trabajo decidimos emplear además los algoritmos Passive Aggressive y Perceptron-Like en el problema de adaptación online a un escenario IMT.

Estos algoritmos necesitan cada valor de KSMR asociado a cada una de los hipótesis para poder identificar la calidad. Dentro de un escenario IMT la lista

de N-best es sustituida por una lista de N-best grafos de palabras. Es necesario utilizar como representación del grafo de palabras el mejor camino de este, es decir, la traducción que da el grafo de palabras para la oración de entrada mediante la cual fue generado. Las estrategias planteadas en la segunda parte de las secciones 2.2.1, 2.2.2 y 2.2.3 se plantean como una alternativa a los algoritmos DRR, PA y PCL definidos para un escenario de post-edición, en donde se aborda el problema de adaptación directamente desde una perspectiva de IMT.

Capítulo 3

Experimentos

En este capítulo del trabajo final de máster se plantean los aspectos más relevantes de los experimentos realizados. Primeramente se describen los corpus empleados, las métricas de calidad utilizadas a lo largo de los experimentos y se detalla la configuración inicial del sistema realizada para el comienzo de los experimentos. Por último se mostrarán los resultados experimentales y las principales conclusiones de los experimentos realizados.

3.1. Corpus

En esta sección se definen lo principales corpus utilizados y sus principales características. Fueron empleados dos corpus un para entrenar el sistema SMT y otro para realizar todas las pruebas. El corpus utilizado para entrenar el Europarl y como corpus de prueba fue empleado el News Commentary. A continuación se realiza una breve descripción de estos corpus y sus principales características.

3.1.1. Europarl

El corpus Europarl [19] está construido a partir de documentos del Parlamento Europeo e incluye versiones en 11 lenguas europeas: románicas (francés, italiano, español y portugués), germánicas (inglés, holandés, alemán, danés y sueco), griego y finlandés. Este corpus fue creado con el objetivo de generar un texto de oraciones alineadas en los diferentes idiomas para sistemas de traducción automática estadística. En la actualidad, el corpus Europarl es referencia en SMT y ha sido utilizado en muchos proyectos. En este trabajo se ha utilizado el corpus Europarl para entrenar nuestro sistema. Las principales características del corpus Europarl pueden verse en el siguiente cuadro:

Cuadro 3.1: Características del corpus Europarl inglés-español utilizado como entrenamiento. De corpus de desarrollo se empleó la unión de las particiones establecidas en las conferencias ACL de los años 2008, 2009 y 2010. "Oraciones" indica el número de oraciones del corpus, "Núm. de Palabras" indica el número de palabras del corpus, "Vocabulario" indica las palabras del vocabulario en el corpus de entrenamiento y "Palabras fueraV" indica palabras fuera del vocabulario del corpus de desarrollo con respecto al Europarl. La letra "k" indica miles de elementos y "M" para millones de elementos.

		ES	EN
Entrenamiento	Oraciones	1.4M	
	Núm. de Palabras	42.3M	37.8M
	Vocabulario	123.4k	115.5k
Desarrollo	Oraciones	7065	
	Núm. de Palabras	42.3M	37.8M
	Palabras fueraV	299	468

3.1.2. News Commentary

El corpus News Commentary está formado a partir de fragmentos de noticias pertenecientes a un dominio diferente al corpus Europarl. Este corpus es muy utilizado para comprobar el rendimiento de los sistemas en tareas de adaptación porque está generado a partir de diferentes fuentes. En el cuadro 3.2 podemos ver las estadísticas de este corpus.

Cuadro 3.2: Características del corpus News Commentary utilizado como prueba (EMNLP 2011). "Oraciones" indica el número de oraciones del corpus, "Núm. de Palabras" indica el número de palabras del corpus, "Long. oraciones" para la longitud media de las oraciones y "Palabras fueraV" indica las palabras fuera del vocabulario con respecto al Europarl. Se utiliza "k" para miles de elementos.

		ES	EN
Prueba	Oraciones	3003	
	Núm. de Palabras	74.7k	79.4k
	Long. de Oraciones	26.5	24.9
	Palabra fueraV	1549	1708

3.2. Métricas de evaluación

Un problema difícil que presenta la traducción automática en la actualidad es la evaluación automática del sistema. En traducción automática una oración de entrada no tiene una oración de salida exclusivamente correcta, puede tener un conjunto de diferentes oraciones correctas como salida. En este trabajo final de máster, se emplean diferentes métricas de calidad para evaluar el rendimiento de los distintos sistemas, son las siguientes:

- KSMR(Keystroke Mouse-action Ratio) [2] KSMR es una métrica de evaluación empleada en el paradigma de IMT. Se calcula contando el número total de pulsaciones de teclado y clicks al ratón que realiza el usuario para lograr la validación de la traducción final, dividido por el número total de caracteres de la oración. En esta métrica se asume que las dos acciones tienen el mismo esfuerzo para el usuario. Se considera una métrica pesimista porque asume una única referencia, mientras que un usuario real puede se más flexible.

- TER (Translation Edit Rate) [41] TER es una métrica de error empleada en traducción automática que mide el número de ediciones necesarias para modificar la salida del sistema de modo que se convierta en la referencia. Se calcula como el mínimo número de ediciones requeridas para modificar las hipótesis del sistema de forma que estas coincidan con la traducción de referencia, normalizado por el número de palabras de la referencia. En este caso, las posibles ediciones incluyen inserciones, borrados, sustituciones por una sola palabra o reordenamiento de secuencias de palabras.

A pesar de emplear diferentes métricas de calidad a lo largo de este trabajo, el objetivo principal es maximizar la calidad de los sistemas en función del Keystroke Mouse-action Ratio (KSMR). A pesar de ello, en IMT también se utilizan otras métricas como el Word Stroke Rate (WSR), a diferencia esta métrica calcula el coste incurrido por el usuario cuando lee el nuevo sufijo proporcionado por el sistema. El motivo por se decidió emplear KSMR en lugar de WSR es porque puede ver cada nuevo carácter para el nuevo sufijo que será mostrado al usuario mientras que WSR sólo se tiene en cuenta el sufijo completado, siendo más fino y permitiendo una mayor granularidad al calcular las nuevas hipótesis. Para el calculo del KSMR se ha empleado el mismo procedimiento que el utilizado en [2].

3.3. Configuración inicial

Para realizar experimentos en un escenario IMT real es necesario disponer de un conjunto de traductores humanos para que interactúen con el sistema

IMT, para corregir cada una de las hipótesis como se muestra en la figura 1.2. Aunque lo planteado anteriormente es lo idóneo resulta complicado emplear traductores humanos para la evaluación del sistema IMT por el alto coste que esto supone, por lo tanto, es necesario emplear una alternativa para simular esta interacción del traductor con el sistema. La alternativa aquí planteada consiste en imitar la evaluación que un traductor humano haría a la hipótesis propuesta por el sistema IMT, utilizando para calcular la calidad de la hipótesis la traducción de referencia del conjunto de prueba. Para hacer esta simulación se empleó el toolkit Thot [31]. Este sistema nos devuelve la simulación de la evaluación que haría un traductor humano, empleando la métrica KSMR.

Para poder distinguir las mejoras que nuestro sistema de adaptación aporta en un escenario IMT lo hemos de comparar con un sistema de referencia. Este sistema de referencia va a ser los resultados obtenidos en el conjunto de prueba con el sistema de simulación de IMT sin aplicarle ningún algoritmo de adaptación.

Para entrenar el sistema SMT se ha usado el toolkit de código abierto Moses, que implantará un sistema de traducción automática [20] basado en segmentos, en su configuración estándar no monótona y estimando $\hat{\boldsymbol{\lambda}}$ usando MERT, empleando como conjunto de desarrollo la unión de las particiones establecidas en los talleres de SMT pertenecientes a la conferencias ACL (Association for Computational Linguistics: Human Language Technologies) del 2008 al 2010.

Moses es un sistema de traducción automática que realiza el entrenamiento de los modelos de traducción para cualquier par de lenguas dado un conjunto de oraciones bilingües. Además, Moses implementa un algoritmo de decodificación que permite encontrar de forma eficiente la traducción más probable para una oración de entrada. Los grafos de segmentos requeridos para llevar a cabo las diversas experimentaciones también se crearon utilizando Moses. Se entrenó un modelo de lenguaje de 5-gramas mediante la herramienta SRILM [42] empleando interpolación y suavizado Kneser-Ney [17].

Además del corpus Europarl se utilizó un conjunto de prueba diferente que no pertenece al dominio del Europarl, News Commentary procedente del taller de traducción automática de la conferencia EMNLP 2011. Este corpus es el que empleamos para poder analizar el rendimiento de las diversas estrategias de adaptación online. Las principales características de los corpus están explicadas en la sección 3.1.

La creación de la lista de N-best grafos de palabras para cada una de las oraciones a traducir del conjunto de prueba a partir de las estrategias planteadas es, con diferencia, el proceso con mayor coste de los experimentos realizados. Esto se debe a que se deben generar las características de cada una de la hipótesis además de simular la interacción del usuario tantas veces como el tamaño de la lista N-best grafos de palabras, multiplicado por el número de

oraciones del conjunto de prueba. Por tanto en este trabajo se ha generado un conjunto de traducciones muy grande, junto con la consecuente generación del grafo de palabras y cálculo del mejor camino, con su posterior cálculo de KSMR.

Todos los experimentos se han realizado dentro del marco IMT, por lo que las oraciones de referencia fueron empleadas para adaptar los parámetros del sistema después de haber realizado la traducción de la oración de entrada y evaluado su calidad.

Por otra parte, los diferentes algoritmos de la sección 2.2 utilizan un cierto número de parámetros libres los cuales han sido establecidos basándose en los experimentos previos de [16] y [37].

- Algoritmo DRR definido para post-edición en la sección 2.2.1 $\alpha = 0,01$ y $\beta = 0,01$.
- Aproximación Gaussiana en la sección 2.3.1: $\beta = 0,01$, muestreo basado en una distribución gaussiana con $\sigma^2 = 0,01$ y $\mu = \hat{\boldsymbol{\lambda}}$, es el conjunto de pesos obtenido empleando la técnica MERT.
- Aproximación Simplex en la sección 2.3.2: $\beta = 0,01$, muestro basado en el algoritmo Simplex, con valores iniciales $\hat{\boldsymbol{\lambda}}$, es el conjunto de pesos obtenido empleando la técnica MERT.

3.4. Resultados experimentales

En esta sección se mostrarán los principales resultados en forma de gráfica o tabla para cada una de las estrategias seguidas para adaptar los pesos del modelo log-lineal en un escenario IMT.

El apartado se encuentra dividido en 4 partes, tres dedicadas a cada una de las estrategias propuestas, es decir adaptación mediante el algoritmo DRR visto en la sección 2.2.1 y definido para un escenario CAT no interactivo, la estrategia aproximación gaussiana definida en la sección 2.3.1 y la aproximación simplex definida en la sección 2.3.2, para cada una de estas dos ultimas aproximaciones se emplearan los tres algoritmos de actualización. Además, hay una sección dedicada al efecto de la poda de los grafos de palabras.

Antes de mostrar los resultados de las distintas estrategias analizamos la importancia del parámetro α, o ratio de aprendizaje, en cada una de ellas. El valor de α puede interpretarse como el encargado de controlar el tamaño del paso de actualización al procesar una muestra, es decir la influencia que tiene la muestra procesada en el valor de los parámetros a predecir. Cuanto menor sea el valor del ratio de aprendizaje la influencia de muestras no representativas es menor, por lo que evita que influyan negativamente en el valor de los parámetros, aunque también disminuye la influencia de las muestras representativas provocando que el proceso de adaptación sea más lento. Por otra parte,

valores altos del ratio de aprendizaje permiten que la adaptación se realice a mayor velocidad, pero causa que el algoritmo sea más sensible. Por lo tanto, seleccionar correctamente el valor adecuado para el ratio de aprendizaje no es una tarea sencilla, y debe realizarse de forma empírica.

A continuación se explica las diferentes gráficas de resultados que se van a presentar en las secciones:

- La primera gráfica para cada sección va a mostrar la influencia del α en los valores de resultado, mediante una métrica de evaluación. Estos resultados serán comparados con el sistema de referencia que permanecerá constante. Se empleará "baseline" para referirnos al sistema de referencia, "DRR" para los resultados empleando el algoritmo Discriminative Ridge Regression, "PA" para los resultados empleando el algoritmo Passive Agressive y "PCL" para los resultados empleando el algoritmo Perceptron-Like.

- En la segunda gráfica se verá evolución de la métrica empleada al procesar progresivamente todas las frases del conjunto de prueba. Se mostrará el valor medio acumulado de la métrica empleada para el subconjunto de oraciones procesadas hasta el momento, se vera tanto para el sistema de referencia como cuando se emplea alguno de los algoritmos de adaptación. Otro aspecto a tener en cuenta es que las curvas de evolución van a ser muy similares entre ellas, esto es debido al corpus de prueba empleado que siempre será el mismo. Cada corpus tiene sus propias características de comportamiento. Esto es debido a la diferencia que existe entre las frases que lo conforman, las frases pueden ser más sencillas o complejas a la hora de traducirlas viéndose reflejado en las curvas de evolución. Por lo tanto, las características de las curvas de evolución no guardan relación con el algoritmo empleado. Se empleará "baseline" para referirnos al sistema de referencia, "DRR" para los resultados empleando el algoritmo Discriminative Ridge Regression, "PA" para los resultados empleando el algoritmo Passive Agressive y "PCL" para los resultados empleando el algoritmo Perceptron-Like.

- Después se presentará la gráfica de aprendizaje donde se refleja la diferencia entre los resultados de los algoritmos medidos por una métrica de evaluación y el resultado del sistema de referencia. Estas gráficas son una ampliación de las gráficas de evolución de métrica porque se pueden ver más detalladamente los resultados. Se mostrarán resultados positivos y negativos de la diferencia entre ellos, cuando el valor es negativo va a significar que el algoritmo logró mejorar el resultado del sistema de referencia, pero si en cambio el resultado es positivo es que el nuestro es mayor porque lo que no se logró mejorar al sistema de referencia. Se em-

pleará “baseline” para referirnos al sistema de referencia, “DRR” para los resultados empleando el algoritmo Discriminative Ridge Regression, “PA” para los resultados empleando el algoritmo Passive Agressive y “PCL” para los resultados empleando el algoritmo Perceptron-Like.

3.4.1. Análisis del factor de poda

Como se planteó anteriormente para los experimentos realizados, el número de traducciones que se hicieron fue muy elevado por lo que el coste de tiempo era bastante grande. Al comenzar la experimentación se estudió el efecto que tiene el tamaño del grafo de palabras en la calidad de la traducción. Los autores de [45] plantean la poda de los grafos de palabras para ver el efecto del tamaño del grafo de palabras. Aplican la poda de los grafos de palabras mediante un umbral $t < 1$ y estudian la variación de la tasa de error. El algoritmo de poda calcula la probabilidad del camino y si este no supera el umbral especificado se descarta. Si el umbral de poda es $t = 0$ el grafo de palabras no sufre ninguna poda y si al contrario el umbral de poda es $t = 1$, el grafo sólo va a contener el mejor camino. En nuestro caso se estudió cómo varía el valor de KSMR de la muestra en el proceso IMT aplicando diferentes umbrales de poda a su grafo de palabras.

Para los experimentos se emplearon 500 grafos de palabras correspondientes a las primeras 500 frases del conjunto de prueba. El conjunto de pesos empleados para la generación de estos grafos de palabras es $\hat{\boldsymbol{\lambda}}$.

En la Figura 3.1 se muestran los resultados de los experimentos. Se muestra la comparación entre: el valor obtenido de KSMR para las 500 frases sin realizarle poda a los grafos de palabras y los valores de KSMR obtenidos en cada experimento con un diferente umbral de poda. Se puede ver que al aumentar el valor del umbral de poda, es decir a medida que eliminamos más caminos del grafo de palabras, el valor de KSMR aumenta con respecto al resultado sin podar. Esto ocurre por estamos caminos de caliada con la poda. Por otra parte con los umbrales de poda más pequeños más cercanos a cero, los resultados están más próximos a cuando no hay poda. Por lo tanto, se decidió emplear grafos de palabras podados en los siguientes experimentos ya que con una adecuada poda del grafo de palabras se pueden obtener resultados similares a cuando no se realiza. Logrando disminuir el tiempo de búsqueda dentro del grafo de palabras en el proceso IMT por ser más pequeños pero de igual calidad al grafo de palabras original.

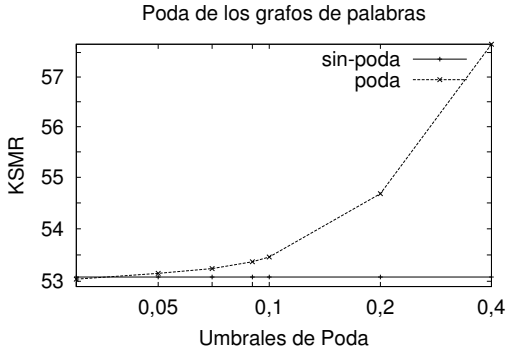


Figura 3.1: Influencia de podar los grafos de palabras en el resultado del proceso IMT medido con la métrica KSMR. La cantidad de grafos de palabras utilizados para los experimentos es 500, correspondientes a las primeras 500 frases de conjunto de prueba.

3.4.2. Minimizando el KSMR mediante el algoritmo DRR definido en escenario post edición

Con los experimentos que se explican en esta sección se tenía como objetivo minimizar el KSMR mediante la minimización por parte del algoritmo DRR descrito en la sección 2.2.1. Primeramente, para probar el funcionamiento del algoritmo DRR definido para post-edición, se creó un sistema SMT básico. Con este sistema fue posible ver la variación del resultado en función del ratio de aprendizaje α.

En [37] se puede observar que al pasar la lista N-best de un valor de N = 5.000 a N = 10.000 no permiten mejorar los resultados del sistema de referencia, por lo que se fija como cantidad recomendada para realizar los experimentos empleando la totalidad del conjunto de prueba un valor de N = 5.000, evitando de esta forma un alto coste computacional y temporal.

La selección del tamaño de la lista de N-best es un aspecto muy importante, ya que influye en dos aspectos:

- La selección de la mejor hipótesis y^*. Es posible que la mejor hipótesis respecto a la referencia $\mathbf{y}^\tau$ dada una lista de N-best no aparezca en una lista de menor tamaño N', $\forall N' < N$, es decir, la hipótesis $\mathbf{y}^*$ para una lista de tamaño N puede no estar entre sus primeras N' posiciones provocando que la hipótesis $\mathbf{y}$ pueda ser distinta para una lista de tamaño N a la de una de tamaño N', y por tanto al aumentar N puede mejorar la calidad de $\mathbf{y}^*$.

- Cuando aumenta el número de posibles hipótesis candidatas la dificultad de la tarea incrementa. Debido a que el sistema SMT produce en orden decreciente de puntuación la lista N-best, conforme aumenta el tamaño de

la lista se aumenta el número de hipótesis de baja calidad e incrementa la diferencia entre el número de *buenas* y *malas* hipótesis.

A continuación, en la figura 3.2 pueden verse los resultados más relevantes de los experimentos realizados. La gráfica de la derecha muestra los resultados de la variación de α en el algoritmo DRR en post-edición empleando la métrica de evaluación TER. Estos resultados se comparan con el resultado del sistema SMT inicial. Como se puede observar los resultados de emplear el algoritmo DRR van mejorando a medida que el valor de α es mayor, obteniéndose el mejor resultado con $\alpha = 0,01$. La causa que para valores de α muy pequeños el sistema empeore es porque hay sobre entrenamiento. La gráfica de la derecha muestra la evolución del TER al procesar todas las muestras del conjunto de prueba, al emplearse el algoritmo DRR en post edición. El valor de α empleado por el algoritmo es 0,01. Podemos ver en la grafica de la izquierda que con ese valor se obtiene el de mejor resultado. Al comenzar el proceso se puede ver que los resultados empleando el algoritmo son peores que el sistema de referencia, pero a partir de la frase 700 aproximadamente los resultados comienzan a ser mejores con respecto al sistema de referencia. Finalmente el valor de TER obtenido con el algoritmo logra mejorar 2 puntos al valor del sistema de referencia.

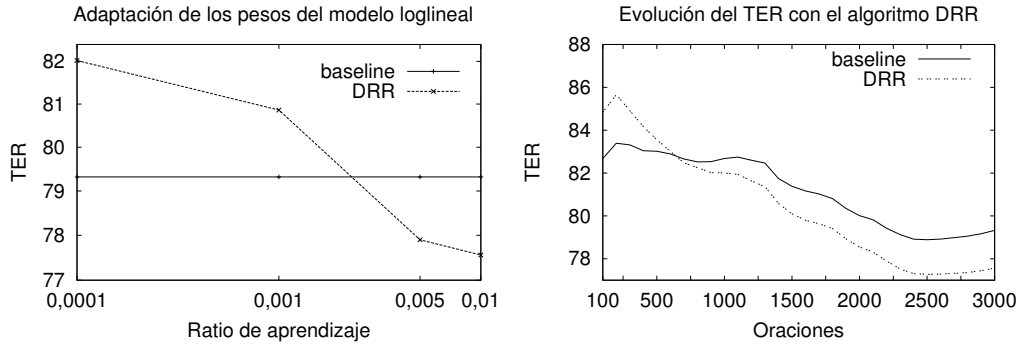


Figura 3.2: Resultados obtenidos en el sistema SMT empleando el algortimo DRR con el conjunto de prueba, con un tamaño de lista N-best N=5.000. La gráfica de la izquierda muestra la influencia de α en los resultados al emplear el algoritmo. La gráfica de la derecha muestra la evolución del TER cuando adaptamos $\boldsymbol{\lambda}$ dentro del conjunto de prueba. El valor de α empleado ha sido de 0,01 que es el valor con el que se obtuvo mejor resultado.

En la figura 3.2 se puede ver cómo el valor de TER mejora notablemente los resultados ofrecidos por el sistema de referencia. Por lo tanto, se intentó optimizar el valor de $\boldsymbol{\lambda}$ a través de la optimización de la métrica de calidad TER mediante el algoritmo DRR definido en la primera parte de la sección 2.2.1, intentando que consecuentemente acabara minimizándose el KSMR.

La gráfica de la figura 3.3 muestra la evolución de la calidad de un sistema IMT medido en KSMR durante el procesado de cada una de las oraciones del conjunto de prueba, empleando los pesos obtenidos mediante el algoritmo DRR dentro del sistema SMT. Para estos experimentos los pesos utilizados son los obtenidos con el valor de α de 0,01, es decir, el valor con el cual se han obtenido los mejores resultados de TER en el sistema SMT. Como puedo observarse, las curvas de evolución del sistema propuesto y el sistema de referencia tienen un comportamiento muy similar. Puede verse la evolución del KSMR, en ambos sistemas el comportamiento es muy similar hasta procesar la oración 1.250 en donde las diferencias comienzan a ser visibles. A partir de dicha oración, el sistema propuesto empleando DRR en post-edición empeora los resultados con respecto al sistema de referencia, es decir, a nuestro sistema le cuesta más encontrar la traducción de las frase por lo que se obtiene un valor de KSMR más elevado.

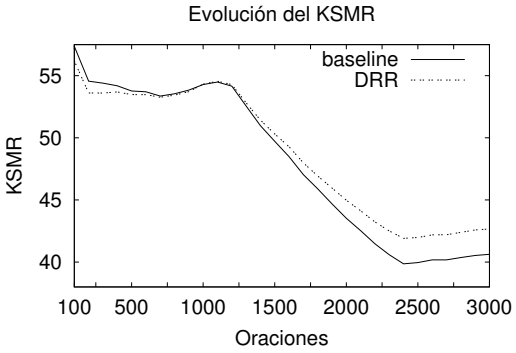
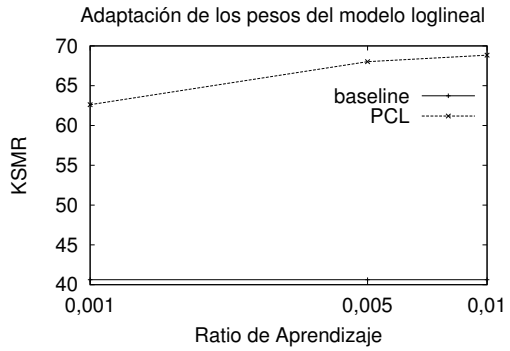


Figura 3.3: Evolución del KSMR cuando adaptamos $\boldsymbol{\lambda}$ dentro del conjunto de prueba. El valor de α empleado ha sido de 0,01. Los resultados han sido generados empleando un sistema IMT simulado.

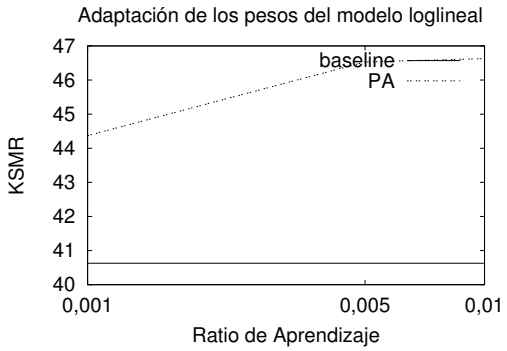
3.4.3. Minimizando el KSMR mediante la aproximación gaussiana

Para intentar minimizar el KSMR en este trabajo se plantearon dos aproximaciones que van a emplear los tres algoritmos de adaptación definidos directamente para IMT. En esta sección se plantean los resultados obtenidos con la aproximación gaussiana para cada uno de los algoritmos. Se ha empleado en todos los experimentos un tamaño máximo de $\boldsymbol{\Lambda} = 201$ conjuntos de pesos, o lo que es lo mismo, de 201 grafos de palabras, incluyendo el grafo de palabras que emplea el conjunto de pesos dado por MERT.

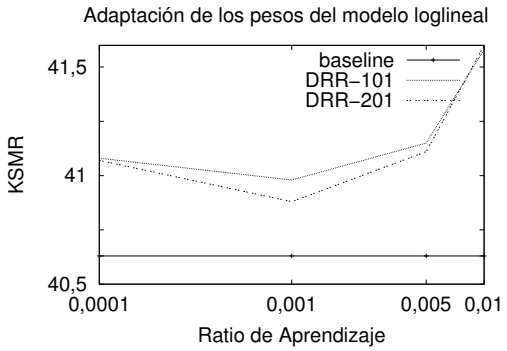
Primeramente se experimentó con la influencia del ratio de aprendizaje con esta aproximación. En la figura 3.4 se muestran los resultados obtenidos para cada uno de los algoritmos al ir variando los valores de α. Para el algoritmo DRR se hicieron varios experimentos según el tamaño $\boldsymbol{\Lambda}$. En esta gráfica se



(a) Algoritmo Perceptron-Like



(b) Algoritmo Passive Agressive

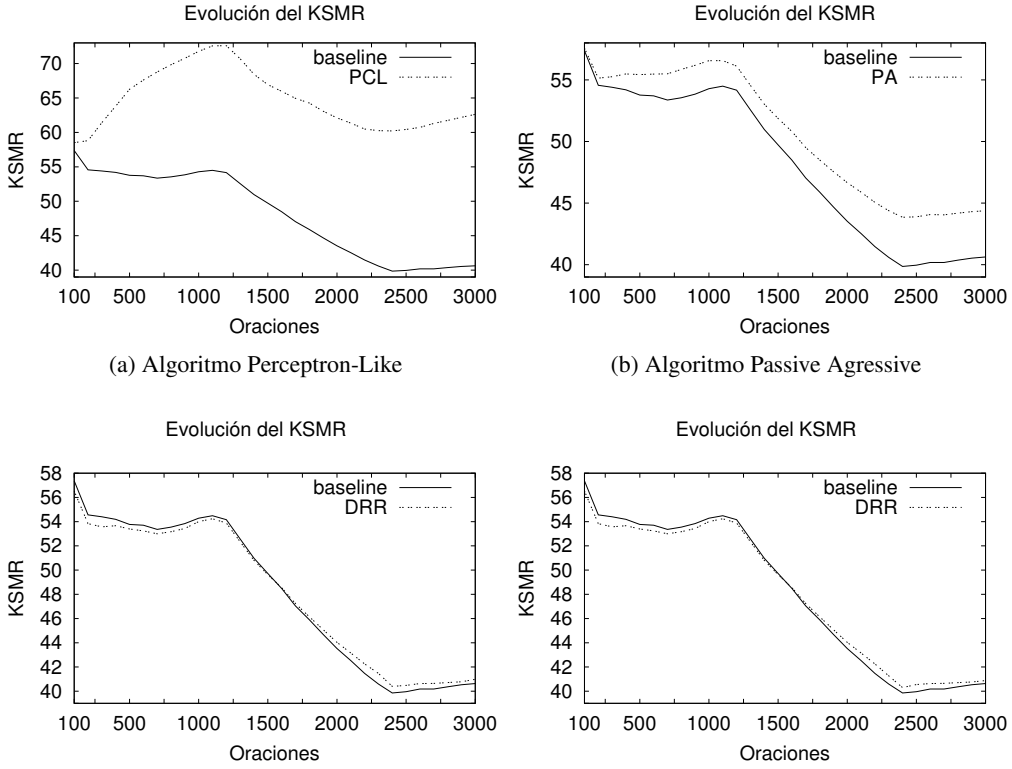


(c) Algoritmo Discriminative Rigde Regression

Figura 3.4: Influencia de α en el rendimiento de los algoritmos para el conjunto de prueba. La cantidad total de conjuntos de pesos utilizados para PA y PCL es 201 incluyendo el conjunto de pesos dado por MERT. Para el algoritmo DRR se muestran los resultados para $\mathbf{\Lambda} = 101$ y $\mathbf{\Lambda} = 201$.

muestran los resultados para $\mathbf{\Lambda} = 101$ y para $\mathbf{\Lambda} = 201$. Para los otros dos algoritmos solamente se realizaron los experimento para $\mathbf{\Lambda} = 201$, esta decisión fue tomada a partir de los resultados obtenidos para el algoritmo DRR que se realizaron antes, en donde los mejores resultados fueron con un $\mathbf{\Lambda} = 201$. Para todos los algoritmos se puede ver que cuando el valor de α es más cercano a 1 el resultado de KSMR también aumenta con respecto al KSMR del sistema de referencia. Esto es producto a que el valor de actualización es mayor, produciendo que el valor de $\boldsymbol{\lambda}_t$ sea muy diferente a $\hat{\boldsymbol{\lambda}}$. Para los algoritmos PA y PCL no se obtienen resultados satisfactorios. Los mejores resultados obtenidos son para un valor de $\alpha = 0{,}001$ para ambos algoritmos, y dichos resultados son mucho peores con respecto al valor de referencia. Para el algoritmo DRR los resultados son mucho más alentadores por lo que se decidió experimentar con un conjunto de α más amplio. El mejor valor de KSMR es para un valor de $\alpha = 0{,}001$ aunque tampoco se logró mejorar el valor de KSMR de referencia.

Después de ver la influencia del ratio de aprendizaje, se puede ver en la figura 3.5 la evolución del KSMR durante el procesado del corpus de prueba para cada uno de los algoritmos de adaptación empleados. Las tres gráficas



(a) Algoritmo Perceptron-Like

(b) Algoritmo Passive Agressive

(c) Algoritmo Discriminative Rigde Regression $\mathbf{\Lambda} = 101$

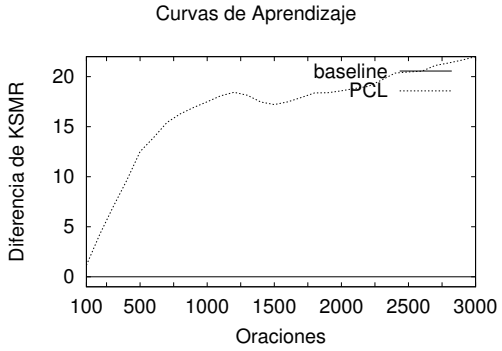
(d) Algoritmo Discriminative Rigde Regression $\mathbf{\Lambda} = 201$

Figura 3.5: Evolución del KSMR cuando adaptamos $\boldsymbol{\lambda}$ dentro del conjunto de prueba, empleando los algoritmos DRR, PA y PCL. Solamente se han dibujado 1 de cada 100 puntos para facilitar la visualización de las gráficas.

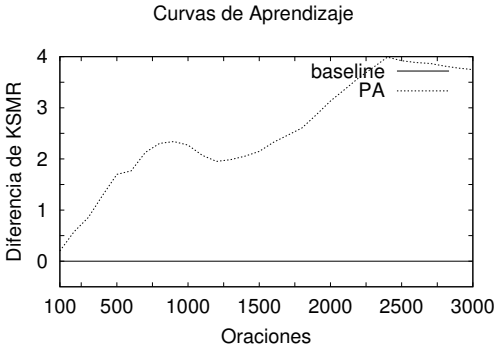
han sido generadas empleando el valor de α con el que cada algoritmo obtuvo su menor valor de KSMR. En la gráfica de algoritmo PCL en todo momento los resultados del algoritmo son peores con respecto al sistema de referencia. Los resultados para el algoritmo PA son un poco mejor con respecto al PCL, pero siguen siendo peores al de referencia. Los mejores resultados obtenidos para esta aproximación es con el algoritmo DRR que como pueden verse en las dos figuras, hasta la oración 1.500 los resultados de KSMR son mejores, a partir de donde comienza a empeorar hasta el final del corpus obteniendo un resultado peor que el de referencia. Esto puede ser debido a que estas primeras 1.500 frases del corpus son más fácil de traducir e influyen positivamente en los parámetros del algoritmo. Se experimentó con diferentes tamaños del conjunto

$\mathbf{\Lambda}$, tal como puede verse en la figura, obteniéndose los mejores resultados con el conjunto de pesos $\mathbf{\Lambda} = 201$.

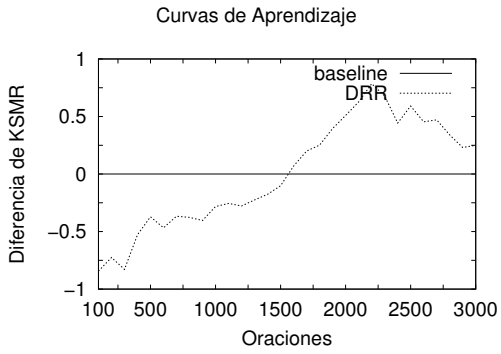
Las curvas de aprendizaje pueden verse en la figura 3.6, donde las tres gráficas muestran las mejoras del sistema IMT medido mediante KSMR. De nuevo para dibujar estas tres gráficas se emplearon como valor α el que obtuvo menor KSMR para cada algoritmo. Puede verse como la diferencia siempre es positiva para los algoritmos PA y PCL por lo que no hubo ningún aprendizaje en todo el corpus. Para el algoritmo DRR la gráfica es negativa para las primeras 1.500 frases, donde comienza a aumentar teniendo un resultado positivo al concluir el corpus por lo que nuestro KSMR es mayor.



(a) Algoritmo Perceptron-Like



(b) Algoritmo Passive Agressive



(c) Algoritmo Discriminative Rigde Regression

Figura 3.6: Curvas de aprendizaje cuando adaptamos $\boldsymbol{\lambda}$ dentro del conjunto de prueba. El empleado es $\alpha = 0,001$ que es con el que se obtuvo mejor resultado.

El cuadro 3.3 sintetiza los resultados más significativos empleando la aproximación gaussiana. Los resultados de este cuadro han sido obtenidos empleando el valor de α que mejores resultados dio en términos de calidad medida mediante KSMR.

Cuadro 3.3: Los resultados han sido obtenidos mediante la aproximación gaussiana. "α" indica el ratio de aprendizaje y "DRR" Discriminative Rigde Regression, "PA" Passive Agressive y "PCL" Perceptron Like.

Algoritmo de adaptación	α	KSMR
Sistema de referencia	-	40,63
DRR $\boldsymbol{\lambda} = 101$	0,001	40,99
DRR $\boldsymbol{\lambda} = 201$	0,001	40,88
PA	0,001	44,37
PCL	0,001	62,62

3.4.4. Minimizando el KSMR mediante la aproximación simplex

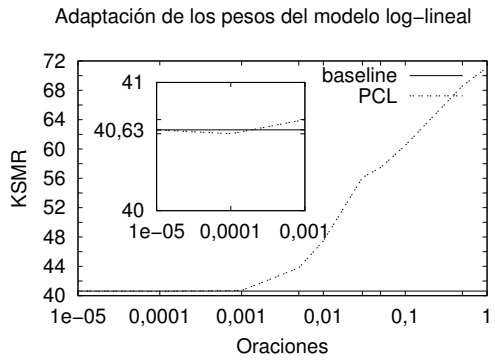
En esta sección se muestran los resultados obtenidos en la aproximación simplex descrita en la sección 2.3.2. Al igual que en la sección anterior, se emplearon los tres algoritmos definidos directamente para IMT.

Como se puede ver, a continuación en la figura 3.7 se muestra la influencia del ratio de aprendizaje en esta estrategia. Las gráficas muestran la calidad del sistema IMT medida en KSMR para cada uno de los algoritmos de adaptación. Aquí se puede ver como a medida que se decrementa el valor de α el KSMR mejora su comportamiento, llegando a mejorar en el algoritmo DRR y PA con respecto a la referencia, aunque de una forma poco significativa. Con un $\alpha = 0,0001$ es donde se obtienen los mejores resultados para estos dos algoritmos, cuando la actualización de los $\boldsymbol{\lambda}$ es sumamente pequeña. En el caso del algoritmo de PCL no se tiene ninguna mejoría al ir variando los valores de α, llegando a emparejarse con el sistema de referencia por lo que no se logra tener aprendizaje.

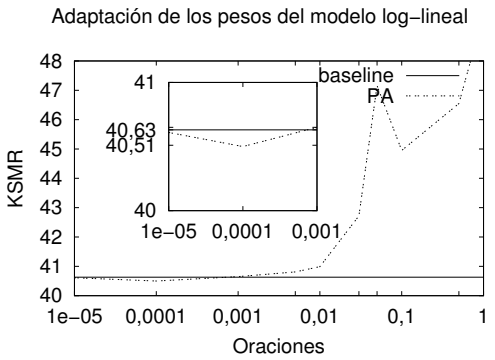
La figura 3.8 muestra la evolución del KSMR, para cada uno de los algoritmos de adaptación utilizados en este trabajo durante el procesado del corpus de prueba. Las tres gráficas han sido generadas empleando un $\alpha = 0,0001$, ya que éste es el valor con el que se obtuvo un menor KSMR.

Las gráficas muestran la evolución del KSMR con el algoritmo DRR y PA. Puede verse como el comportamiento de ambos es casi idéntico al del sistema de referencia. Esto se debe a que la actualización de los parámetros del modelo log-lineal es de una forma muy sutil, provocando que apenas se pueda ver una mejora significativa de dos décimas en las últimas 500 frases. En el caso del algoritmo de adaptación PCL la evolución del KSMR es casi idéntica al del sistema de referencia y el resultado final del valor de KSMR es menor a media décima con respecto al valor del sistema de referencia.

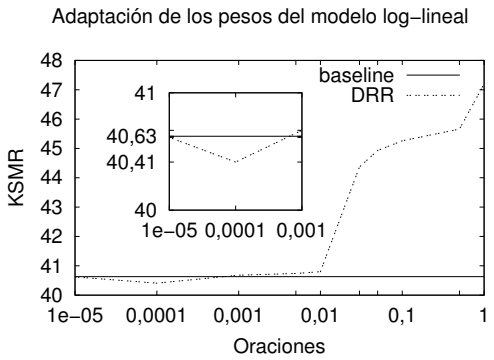
La figura 3.9 muestra las curvas de aprendizaje del sistema IMT adaptado y del sistema de referencia, en cada una de las tres gráficas se puede ver el rendimiento de cada uno de los algoritmos en función de la métrica de calidad



(a) Algoritmo Perceptron-Like



(b) Algoritmo Passive Agressive

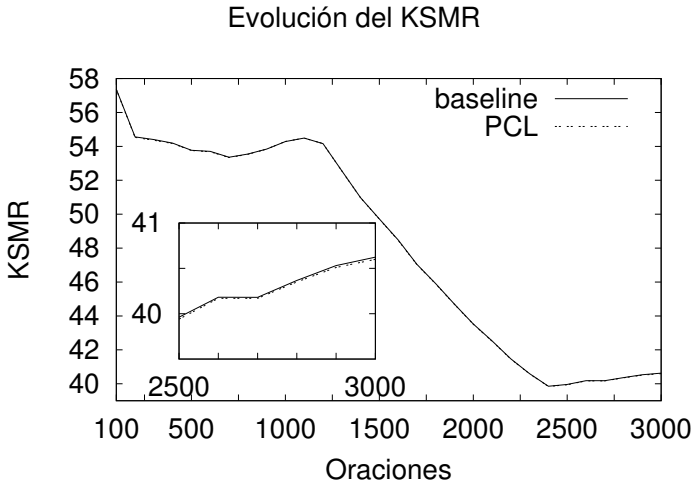


(c) Algoritmo Discriminative Rigde Regression

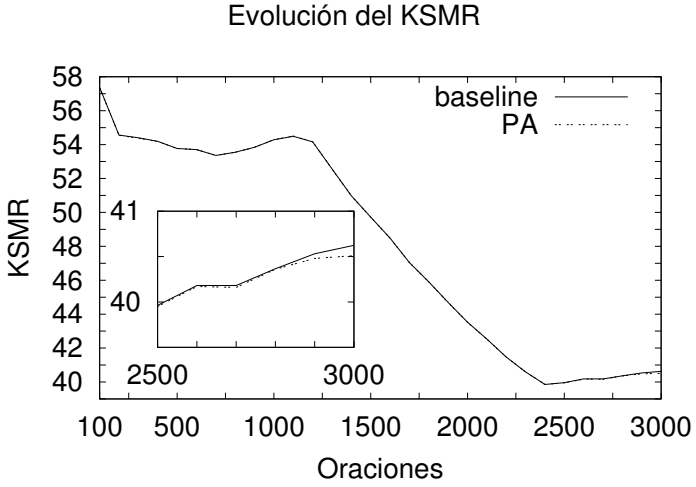
Figura 3.7: Influencia de α en el rendimiento de los algoritmos para el conjunto de prueba. La cantidad de conjuntos de pesos utilizados fueron los obtenidos mediante el método simplex.

KSMR. Para estas tres gráficas, el valor de α empleado ha sido de nuevo de 0,0001 ya que es el valor con el cual se han obtenido los mejores resultados de KSMR. En la gráfica del algoritmo PCL se puede ver como la diferencia entre los dos sistemas es escasa. Para los otros dos algoritmos se puede observar para casi todas las muestras del conjunto de prueba que el sistema IMT adaptado mejora respecto al de referencia, y al acabar, la mejora lograda es de aproximadamente dos décimas para el algoritmo DRR y una décima para el algoritmo PA.

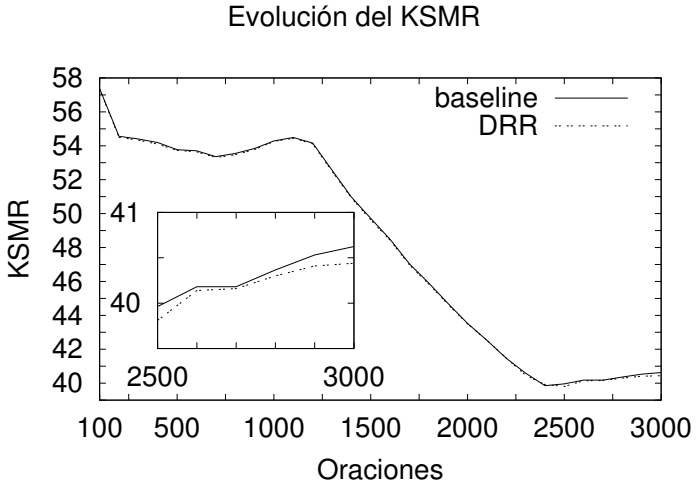
El cuadro 3.4 sintetiza los resultados más significativos empleando la aproximación simplex. Los resultados de este cuadro han sido obtenidos empleando el valor de α que mejores resultados dio en términos de calidad medida mediante KSMR, que en este caso fue 0,0001.



(a) Algoritmo Perceptron-Like

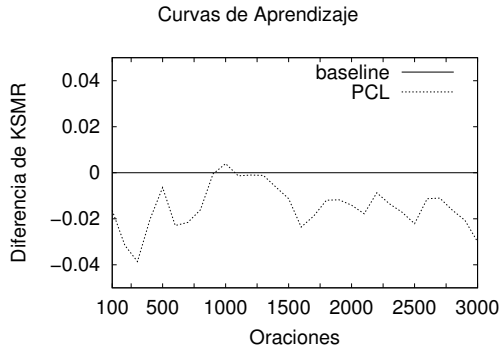


(b) Algoritmo Passive Agressive

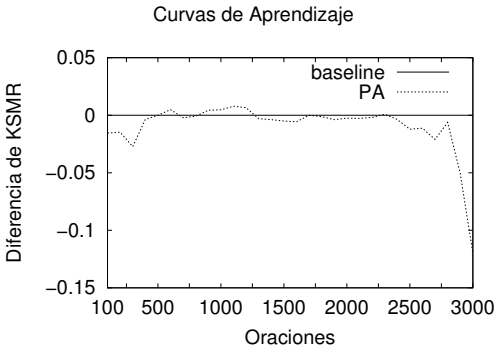


(c) Algoritmo Discriminative Rigde Regression

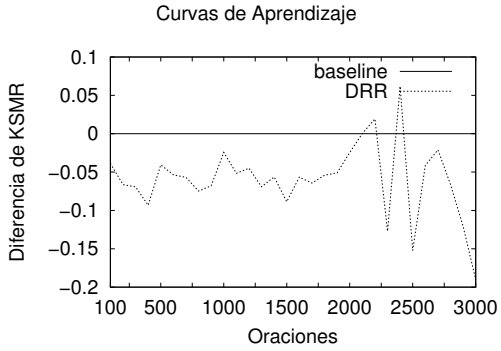
Figura 3.8: Evolución del KSMR cuando adaptamos $\boldsymbol{\lambda}$ dentro del conjunto de prueba, empleando los algoritmos DRR, PA y PCL. Solamente se han dibujado 1 de cada 100 puntos para facilitar la visualización de las gráficas. El valor de α empleado ha sido de 0,0001.



(a) Algoritmo Perceptron-Like



(b) Algoritmo Passive Agressive



(c) Algoritmo Discriminative Rigde Regression

Figura 3.9: Curvas de aprendizaje cuando adaptamos $\boldsymbol{\lambda}$ dentro del conjunto de prueba

Cuadro 3.4: Los resultados han sido obtenidos mediante la aproximación simplex. "α" indica el ratio de aprendizaje y "DRR" Discriminative Rigde Regression, "PA" Passive Agressive y "PCL" Perceptron Like.

Algoritmo de adaptación	α	KSMR
Sistema de referencia	-	40,63
DRR	0,0001	40,44
PA	0,0001	40,51
PCL	0,0001	40,60

3.5. Análisis de ejemplos

En la presente sección vamos hacer un análisis del mejor camino del grafo de palabras, es decir, nos referimos a la hipótesis inicial que va proponer el sistema IMT simulado. En este caso solamente se va a realizar este análisis para la aproximación simplex porque es con la que se obtuvo los mejores resultados.

En el cuadro 3.5 se puede ver las hipótesis iniciales propuestas por el sistema IMT simulado. Para la obtención de los pesos se siguió la aproximación

simplex. Cada uno de los ejemplos se corresponde a uno de los algoritmos de adaptación empleados.

Cuadro 3.5: Comparación de tres traducciones generadas por el sistema IMT de referencia y el sistema IMT adaptado siguiendo la aproximación simplex. "Int." indica el número de interacciones necesarias para convertir la oración dada por el sistema en la oración de referencia en un sistema IMT, "Oración" es la oración de entrada del sistema $\mathbf{x}$, "Referencia" se corresponde con y^{τ} , "Sistema Referencia" es la oración obtenida del sistema de referencia y "DRR" indica que es la oración obtenida empleando el algoritmo Discriminative Rigde Regression, "PA" para el algoritmo Passive Agressive y "PCL" para el algoritmo Perceptron Like. vista en la sección

Algoritmos	**Oraciones**	**Int.**
Oración	architect Josep Maria Jujol cooperated with him .	-
Referencia	colaboró en el diseño el arquitecto Josep María Jujol .	-
Sistema Referencia	el arquitecto Josep Maria Jujol colaboraron con él .	29
DRR	el arquitecto Josep Maria Jujol colaboraron con él .	24
Oración	a suitable size of 50 x 60 CMIS recommended , preferred for example in Scandinavia .	-
Referencia	el tamaño adecuado recomendado es de 50 x 60 cm , algo normal por ejemplo en Escandinavia .	-
Sistema Referencia	un tamaño adecuado de 50 x 60 CMIS recomendó , preferido por ejemplo en Escandinavia .	42
PA	un tamaño adecuado de 50 x 60 CMIS recomendó , preferido por ejemplo en Escandinavia .	42
Oración	the Red Bull hero , without any doubt , is the recent champion Sebastian Vetell .	-
Referencia	el héroe de la Fórmula 1 de Red Bull , sin duda , es el reciente campeón Sebastian Vetell .	-
Sistema Referencia	la Red Bull héroe es , sin duda alguna , el reciente campeón Sebastian Vetell .	21
PCL	la Red Bull héroe es , sin duda alguna , el reciente campeón Sebastian Vetell .	25

Como puede verse en el primer ejemplo, la hipótesis generada por el sistema adaptado coincide con la dada por el sistema de referencia. A pesar de que las hipótesis propuestas coinciden, el esfuerzo necesario para corregir la hipótesis generada por el sistema de adaptación es menor que el necesario para corregir la oración propuesta por el sistema de referencia. Esto se debe a que al haber modificado las puntuaciones de las aristas de los grafos de palabras mediante el algoritmo de adaptación online empleado, el mejor camino es el mismo pero el grafo resultante es mejor.

En el segundo ejemplo no se logra encontrar mejores sufijos en las posterio-

res interacciones con el humano, por lo tanto, no se logra disminuir el esfuerzo necesario para corregir la traducción. En el tercer ejemplo las traducciones generadas por el sistema IMT y el sistema de referencia coinciden pero en este caso el número de interacciones que se necesita en el sistema IMT adaptado debido a las modificaciones de las puntuaciones de las aristas de grafo empeoraron la calidad del grafo.

3.6. Conclusiones

En este capítulo se han podido ver los resultados de las dos aproximaciones planteadas para la generación de los pesos, empleando cada una los tres algoritmos de adaptación redefinidos para IMT. Los primeros experimentos se realizaron para ver el comportamiento del algoritmo Discriminative Rigde Regression en un escenario de post-edición para intentar minimizar la métrica de calidad TER y con los mejores resultados tratar de optimizar el KSMR, asumiendo que existe una correlación entre el KSMR y el TER. Con esto solo se logró mejorar el TER pero no se obtuvieron resultados positivos para el KSMR.

Otro aspecto que se tuvo en cuenta en los primeros experimentos en escenario IMT es el tamaño del grafo de palabras. Se realizaron diferentes experimentos variando el umbral de poda obteniendo valores diferentes y comprobando el efecto que tiene el tamaño del grafo de palabras en el resultado del proceso IMT.

La aproximación gaussiana fue definida para generar los conjuntos de pesos de forma semi-aleatoria mediante distribuciones gaussianas. Con esta aproximación se emplearon los tres algoritmos de adaptación definidos para IMT. Los resultados obtenidos para los algoritmos Passive Agressive y Percetron-Like no mejoraron con ningún valor de α al sistema de referencia. Para el algoritmo Discriminative Rigde Regression los resultados fueron un poco mejores pero tampoco se logró disminuir el valor del KSMR con respecto al de referencia. Una de las causas de estos resultados con respecto a los obtenidos en [37] es debido al tamaño de $\mathbf{\Lambda}$ que en su caso es de 501 vectores de pesos muy superior al utilizado en nuestro caso que es 201. No se amplió el tamaño de $\mathbf{\Lambda}$ por falta de tiempo y porque se tenía interés en explorar el método simplex.

Por otra parte se realizaron los experimentos para la aproximación simplex. En este caso los conjuntos de pesos se obtenían mediante el algoritmo simplex. Nuevamente se emplearon los tres algoritmos. La mejora del KSMR con esta estrategia es muy pequeña con respecto al sistema de referencia para un valor de α muy pequeño. Esto puede deberse principalmente a que hay poca variación de los conjuntos de pesos obtenidos mediante al algoritmo simplex. El algoritmo puede caer en mínimos locales por lo que los pesos son similares entre ellos.

Capítulo 4

Conclusiones

4.1. Conclusiones

En este trabajo final de máster se desarrollaron estrategias de aprendizaje online de los pesos del modelo log-lineal dentro de un escenario IMT simulado. Para esto se analizó la aplicación de los algoritmos Discriminative Ridge Regression, Passive Agressive y Perceptron-Like en un escenario IMT, debido a su éxito en adaptación para sistemas SMT.

El algoritmo DRR en su definición para un escenario de post-edición fue empleado para actualizar los pesos del modelo log-lineal dentro de un sistema SMT. Empleando posteriormente los pesos obtenidos para la adaptación del sistema IMT. A pesar del éxito del algoritmo DRR en SMT esta definición falla dentro de un escenario IMT, no ofreciendo los resultados esperados. Por ello se decidió emplear los tres algoritmos en un escenario IMT teniendo que plantear nuevas definiciones.

A partir de estas nuevas definiciones para los algoritmos fue necesario replantearse el método de reordenación de las hipótesis. No se reordenaron las diferentes hipótesis para una oración de entrada $\mathbf{x}$, sino que se reordenaron los distintos grafos de palabras pertenecientes a una misma oración. Esta reordenación se realizó empleando como representación del grafo de palabras el mejor camino de este. Para tener variedad de grafos de palabras de una oración de entrada, es necesario disponer de diferentes conjuntos de pesos pero que no sean obtenidos de manera aleatoria. Para la generación de estos conjuntos de pesos se plantearon dos aproximaciones diferentes. La primera, a la que denominamos aproximación gaussiana, obtiene los pesos de forma semi-aleatoria mediante el muestreo de distribuciones gaussianas. La segunda de estas aproximaciones se basa en el algoritmo simplex para la obtención de los conjuntos de pesos y la llamamos aproximación simplex.

Se realizó un conjunto de experimentos para cada uno de los algoritmos de adaptación redefinidos para IMT, donde se empleaba una aproximación de las definidas para obtener los conjuntos de pesos.

Los resultados obtenidos para los algoritmos empleando la aproximación gaussiana no fueron los esperados, a diferencia de los obtenidos en [37] donde se emplea el algoritmo Discriminative Rigde Regression logrando resultados mejores que en nuestro caso. La causa de esto es debido a la cantidad de conjuntos de pesos empleados, que en [37] fue de 501 conjuntos, mientras que en este trabajo se emplearon 201 como máximo. El motivo de probar con esta cantidad de conjuntos pesos y no ampliarlo es por el interés que existía en analizar el comportamiento de los algoritmos planteados con la otra aproximación desarrollada.

En el caso de los experimentos empleando la aproximación simplex se logran resultados más alentadores para los tres algoritmos aunque no se llega a obtener resultados tan notables como los vistos en [16] en un escenario de post-edición y bajo su definición original. Estos resultados parecen indicar que esta estrategia puede llevar a mejorías, aunque se hace necesario un estudio más intensivo. Además, si se comparan con los resultados obtenidos en [37] con los presentados en este trabajo, las mejoras obtenidas son muy similares (si bien mediante una aproximación distinta), a pesar de que con la aproximación simplex el número de conjuntos de pesos es aproximadamente 70 contra 501 que es la cantidad empleada en [37].

Para futuros trabajos, nos gustaría emplear estos algoritmos para la actualización de los pesos que intervienen directamente en el proceso IMT, a diferencia de los modificados en este trabajo que son los que se emplean para la generación del grafo de palabras, lo que podría influir de manera más directa en el proceso IMT. Por ejemplo, los pesos que intervienen en el modelo corrector de errores, esperando obtener mejores resultados.

En la última sección del capítulo de experimentos se muestran ejemplos de las hipótesis iniciales del proceso IMT y puede verse que a partir de una misma hipótesis pueden derivar diferentes resultados. Esto es debido a los cambios que el proceso de aprendizaje online realiza en los grafos de palabras. Por tanto, sería de interés abordar el tema de representar el grafo de palabras de un modo diferente al empleado en este trabajo que es mediante su mejor camino.

Bibliografía

[1] D. Arnold. *Machine translation: an introductory guide*. Cleary Business Studies. NCC Blackwell, 1994.

[2] S. Barrachina, O. Bender, F. Casacuberta, J. Civera, E. Cubel, S. Khadivi, A. Lagarda, H. Ney, J. Tomás, E. Vidal, and J.-M. Vilar. Statistical approaches to computer-assisted translation. Number 35(1), pages 3–28.

[3] Peter F. Brown, Stephen A. Della Pietra, Vincent J. Della Pietra, Robert L. Mercer, and Surya Mohanty. Dividing and conquering long sentences in a translation system. In *Proceedings of the Speech and Natural Language Workshop*, pages 267–271, 1992.

[4] Peter F Brown, Vincent J Della Pietra, Stephen A Della Pietra, and Robert L Mercer. The mathematics of statistical machine translation: Parameter estimation. *Computational linguistics*, 19(2):263–311, 1993.

[5] P.F. Brown, J. Cocke, S.A. Della Pietra, F. Jeninek, J.D. Lafferty, R.L. Mercer, and P.S. Roosin. A Statistical Approach to Machine Translation. *Computational Linguistics*, 16(2):79–85, 1990.

[6] Chris Callison-Burch, Colin Bannard, and Josh Schroeder. Improved statistical translation through editing. In *EAMT-2004 Workshop*, 2004.

[7] Chris Callison-Burch, Cameron Fordyce, Philipp Koehn, Christof Monz, and Josh Schroeder. (meta-) evaluation of machine translation. In *Proceedings of the Second Workshop on Statistical Machine Translation*, StatMT '07, pages 136–158, Stroudsburg, PA, USA, 2007. Association for Computational Linguistics.

[8] Chris Callison-Burch, Philipp Koehn, Christof Monz, and Omar F. Zaidan. WMT '11. 2011.

[9] Francisco Casacuberta, Jorge Civera, Elsa Cubel, Antonio L Lagarda, Guy Lapalme, Elliott Macklovitch, and Enrique Vidal. Human interaction for high-quality machine translation. *Communications of the ACM*, 52(10):135–138, 2009.

[10] Michael Collins. Discriminative training methods for hidden markov models: theory and experiments with perceptron algorithms. In *Proceedings of the ACL-02 conference on Empirical methods in natural language processing - Volume 10*, EMNLP '02, pages 1–8, Stroudsburg, PA, USA, 2002. Association for Computational Linguistics.

[11] Koby Crammer, Ofer Dekel, Joseph Keshet, Shai Shalev-Shwartz, and Yoram Singer. Online passive-aggressive algorithms. *J. Mach. Learn. Res.*, 7:551–585, December 2006.

[12] Arthur P Dempster, Nan M Laird, and Donald B Rubin. Maximum likelihood from incomplete data via the em algorithm. *Journal of the Royal Statistical Society. Series B (Methodological)*, pages 1–38, 1977.

[13] Cristina España Bonet, Lluís Màrquez Villodre, et al. Robust estimation of feature weights in statistical machine translation. 2010.

[14] Guillem Gascó, Martha-Alicia Rocha, Germán Sanchis-Trilles, Jesús Andrés-Ferrer, and Francisco Casacuberta. Does more data always yield better translations? In *Proceedings of the 13th Conference of the European Chapter of the Association for Computational Linguistics*, EACL '12, pages 152–161, Stroudsburg, PA, USA, 2012. Association for Computational Linguistics.

[15] Jesús Tomás Gironés and Francisco Casacuberta Nolla. *Traducción automática de textos entre lenguas similares utilizando métodos estadísticos*. Universidad Politécnica de Valencia, 2003.

[16] Pascual Martínez Gomez. Online learning via dynamic reranking for computer assisted translation. Master's thesis, Universidad Politécnica de Valencia, España, 2010.

[17] Reinhard Kneser and Hermann Ney. Improved backing-off for m-gram language modeling. In *Acoustics, Speech, and Signal Processing, 1995. ICASSP-95., 1995 International Conference on*, volume 1, pages 181–184. IEEE, 1995.

[18] Philipp Koehn. *Statistical Machine Translation*. Cambridge University Press.

[19] Philipp Koehn. Europarl: A parallel corpus for statistical machine translation. In *MT summit*, volume 5, 2005.

[20] Philipp Koehn, Hieu Hoang, Alexandra Birch, Chris Callison-Burch, Marcello Federico, Nicola Bertoldi, Brooke Cowan, Wade Shen, Christine Moran, Richard Zens, Chris Dyer, Ondřej Bojar, Alexandra Constantin, and Evan Herbst. Moses: open source toolkit for statistical machine translation. In *Proceedings of the 45th Annual Meeting of the ACL on Interactive Poster and Demonstration Sessions*, ACL '07, pages 177–180, Stroudsburg, PA, USA, 2007. Association for Computational Linguistics.

[21] Philippe Langlais and Guy Lapalme. Trans type: Development-evaluation cycles to boost translator's productivity. *Machine Translation*, 17(2):77–98, September 2002.

[22] Daniel Marcu and William Wong. A phrase-based, joint probability model for statistical machine translation. In *Proceedings of the ACL-02 conference on Empirical methods in natural language processing-Volume 10*, pages 133–139. Association for Computational Linguistics, 2002.

[23] Pascual Martínez-Gómez, Germán Sanchis-Trilles, and Francisco Casacuberta. Online adaptation strategies for statistical machine translation in post-editing scenarios. *Pattern Recognition*, 45(9):3193–3203, 2012.

[24] Robert C. Moore and William Lewis. Intelligent selection of language model training data. In *Proceedings of the ACL 2010 Conference Short Papers*, ACLShort '10, pages 220–224, Stroudsburg, PA, USA, 2010. Association for Computational Linguistics.

[25] Lewis M.P. Ethnologue: Languages of the world. 2009.

[26] Makoto Nagao. A framework of a mechanical translation between japanese and english by analogy principle. 1984.

[27] John A Nelder and Roger Mead. A simplex method for function minimization. *The computer journal*, 7(4):308–313, 1965.

[28] Franz Josef Och. Minimum error rate training in statistical machine translation. In *Proceedings of the 41st Annual Meeting on Association for Computational Linguistics-Volume 1*, pages 160–167. Association for Computational Linguistics, 2003.

[29] Franz Josef Och and Hermann Ney. Discriminative training and maximum entropy models for statistical machine translation. In *Proceedings of the 40th Annual Meeting on Association for Computational Linguistics*, pages 295–302. Association for Computational Linguistics, 2002.

[30] Franz Josef Och, Richard Zens, and Hermann Ney. Efficient search for interactive statistical machine translation. In *Proceedings of the tenth conference on European chapter of the Association for Computational Linguistics - Volume 1*, EACL '03, pages 387–393, Stroudsburg, PA, USA, 2003. Association for Computational Linguistics.

[31] D. Ortiz-Martínez, I. García-Varea, and F. Casacuberta. Thot: a toolkit to train phrase-based statistical translation models. In *Tenth Machine Translation Summit*. AAMT, Phuket, Thailand, September 2005.

[32] Daniel Ortiz-Martínez, Ismael García-Varea, and Francisco Casacuberta. Online learning for interactive statistical machine translation. In *Human Language Technologies: The 2010 Annual Conference of the North American Chapter of the Association for Computational Linguistics*, HLT '10, pages 546–554, Stroudsburg, PA, USA, 2010. Association for Computational Linguistics.

[33] Kishore Papineni, Salim Roukos, Todd Ward, and Wei-Jing Zhu. Bleu: a method for automatic evaluation of machine translation. In *Proceedings of the 40th Annual Meeting on Association for Computational Linguistics*, ACL '02, pages 311–318, Stroudsburg, PA, USA, 2002. Association for Computational Linguistics.

[34] Kishore A Papineni, Salim Roukos, and R Todd Ward. Maximum likelihood and discriminative training of direct translation models. In *Acoustics, Speech and Signal Processing, 1998. Proceedings of the 1998 IEEE International Conference on*, volume 1, pages 189–192. IEEE, 1998.

[35] Michael JD Powell. An efficient method for finding the minimum of a function of several variables without calculating derivatives. *The computer journal*, 7(2):155–162, 1964.

[36] Frank Rosenblatt. The perceptron: a probabilistic model for information storage and organization in the brain. *Psychological review*, 65(6):386, 1958.

[37] Francisco Javier López Salcedo. Aprendizaje online de los pesos del modelo log-lineal en traducción automática interactiva. Master's thesis, Universidad Politécnica de Valencia, España, 2012.

[38] Holger Schwenk and Jean Senellart. Translation model adaptation for an arabic/french news translation system by lightly-supervised training. 2009.

[39] Kashif Shah, Loïc Barrault, and Holger Schwenk. Translation model adaptation by resampling. In *Proceedings of the Joint Fifth Workshop on Statistical Machine Translation and MetricsMATR*, pages 392–399. Association for Computational Linguistics, 2010.

[40] Claude Elwood Shannon and Warren Weaver. A mathematical theory of communication, 1948.

[41] Matthew Snover, Bonnie Dorr, Richard Schwartz, Linnea Micciulla, and John Makhoul. A study of translation edit rate with targeted human annotation. In *Proceedings of association for machine translation in the Americas*, pages 223–231, 2006.

[42] Andreas Stolcke. Srilm-an extensible language modeling toolkit. In *Seventh International Conference on Spoken Language Processing*, 2002.

[43] Jesús Tomás and Francisco Casacuberta. Monotone statistical translation using word groups. *Procs. of the Machine Translation Summit VIII*, pages 357–361, 2001.

[44] Germán Sanchis Trilles. *Building task-oriented machine translation systems*. PhD thesis, Universitat Politècnica de València, 2012.

[45] Nicola Ueffing, Franz Josef Och, and Hermann Ney. Generation of word graphs in statistical machine translation. In *Proceedings of the ACL-02 conference on Empirical methods in natural language processing - Volume 10*, EMNLP '02, pages 156–163, Stroudsburg, PA, USA, 2002. Association for Computational Linguistics.

[46] B. Vauquois. A survey of formal grammars and algorithms for recognition and transformation in mechanical translation., 1968.

[47] W Weaver and Locke W. Machine translation of languages. In *MIT Press*, pages 15–23, 1945.

[48] Richard Zens and Hermann Ney. Improvements in phrase-based statistical machine translation. In *Proceedings of HLT-NAACL 2004*, pages 257–264. Boston, MA, 2004.

[49] Richard Zens, Franz Josef Och, and Hermann Ney. Phrase-based statistical machine translation. In *KI 2002: Advances in Artificial Intelligence*, pages 18–32. Springer, 2002.

Listado de Símbolos y Abreviaturas

Abreviatura	Descripción	Definición
SMT	Statistical Machine Translation	1.1
CAT	Computer Assisted Translation	1.3
IMT	Interactive Machine Translation	1.4
$\mathbf{x}$	oración de entrada	page 3
$\mathbf{y}$	oración de salida	page 3
$\hat{\mathbf{y}}$	oración estimada	page 3
PB	phrase-based	1.2
M	número de modelos log-lineal	page 5
$h_m(\mathbf{x}, \mathbf{y})$	función de pesos	page 5
λ_m	vector de pesos para $h_m(\mathbf{x}, \mathbf{y})$	page 6
$s(\mathbf{x}, \mathbf{y})$	valor para la hipótesis	page 5
$\mathbf{h}(\cdot\|\cdot)$	vector de características	page 5
$\boldsymbol{\lambda}$	vector de pesos	page 5
KSMR	métrica de precisión	page 29
TER	métrica de error	page 29
DRR	Discriminative Rigde Regression	2.2.1
PA	Passive Aggressive	2.2.2
PCL	Perceptron-Like	2.2.3
Mert	minimum error rate training	page 6
$\hat{\lambda}$	conjunto de pesos obtenidos en la etapa de tunning con la técnica del MERT, son nuestros pesos baseline	page 23
$\mathbf{y}^*$	mejor hipótesis	page 16
p	prefijo en el proceso IMT	page 10
S_h	sufijo en el proceso IMT	page 9
S_l	sufijo incorrecto en el proceso IMT	page 10
k	nueva palabra introducida por el usuario en el proceso IMT	page 10
α	taza de aprendizaje	page 18
$\check{\boldsymbol{\lambda}}_t$	Vector de pesos estimado en tiempo t	page 19

Abreviatura	Descripción	Definición
C	factor de agresividad	page 21
$\phi(\hat{y})$	diferencia entre valores	page 21
$\mathrm{H}_{\mathbf{x}}$	matriz de valores	page 18
$\mathrm{H}^{*}_{\mathbf{x}}$	matriz de los mejores valores	page 19
$\mathrm{R}_{\mathbf{x}}$	matriza de diferencias de valores	page 19
$\mathbf{l}_{\mathbf{x}}$	vector columna de las diferencias entre las medidas de calidad	page 19
I	matriz identidad	page 19
β	termino de regularización	page 19
μ	conjunto de medidas de calidad	page 17
$l(\mathbf{y})$	diferencia entre las medidas de calidad	page 17
λ^{n}	conjunto de pesos obtenidos de forma semi-aleatoria	page 20
$W_{\lambda^{n}}(x)$	grafo de palabras a partir de un conjunto de pesos λ^{n}	page 20
Λ	vector de conjuntos de pesos obtenidos de forma semi-aleatoria	page 20
Moses	decoder	page 30
Thot	toolkit empleado para la simulación del escenario IMT	page 30
ACL	Association for Computational Linguistics	page 30

Índice de figuras

Índice de cuadros