UNIVERSITAT POLITÈCNICA DE VALÈNCIA

DSIC
DEPARTAMENT DE SISTEMES INFORMÀTICS I COMPUTACIÓ

Departamento de sistemas informáticos y computación

Trabajo final Traducción Automática

Traducción automática

Máster Universitario en Inteligencia Artificial, Reconocimiento de Formas e Imagen Digital

Autor
Francisco Javier Gil-Terrón Rodríguez

2021 - 2022

Tabla de contenidos

1. Introducción

El presente trabajo consiste, en esencia, en construir el mejor traductor posible del inglés al español para un subconjunto del corpus Europarl-v7 compuesto por 35.000 frases en el caso del conjunto de entrenamiento, y 1.000 para el de test. Para ello se empleará, principalmente, sistemas de traducción automática estadística con Moses y basado en redes neuronales recurrentes con NMT-Keras, de manera similar a como ya se hizo durante las prácticas.

Para las pruebas con Moses se ha trabajado sobre Polilabs, ya que no se consiguió hacer funcionar Moses sobre docker.

En el otro caso, el de NMT-Keras, se ha utilizado, al igual que para el desarrollo de la respectiva práctica, una máquina Windows y docker con una GPU NVIDIA GeForce GTX 1060 de 6GB.

Para ambos, y en todos los experimentos llevados a cabo a lo largo de este trabajo, se ha utilizado el conjunto de entrenamiento entero (las 35.000 frases) dividiéndolo en 33.500 para entrenamiento y 1.500 para desarrollo.

Las pruebas que se han realizado siguen la misma estructura que se proponía en el boletín de prácticas con el fin de encontrar el mejor modelo posible y únicamente se han omitido aquellas pruebas que no mejorarían los resultados a ciencia cierta, como estudiar la variación de BLEU en función del nº de iteraciones de MERT y asegurando que converja siempre; mientras que se han repetido pruebas realizadas durante las prácticas a pesar de obtener malos resultados por la posibilidad de que alcanzaran unas buenas prestaciones con el corpus actual.

Finalmente, también se realizarán pruebas con modelos de Transformer de NMT-Keras y se utilizará adicionalmente el toolkit OpenNMT.

2. Ejercicios básicos

2.1 Traducción con Moses

Se comenzará tokenizando el corpus y limpiando los datos para ambos idiomas de manera que las frases queden alineadas.

Los siguientes pasos serán idénticos a los que se llevaron a cabo en la práctica. En primer lugar, se entrena el modelo de lenguaje para el español en este caso utilizando SRILM, se obtienen los modelos de reordenamiento empleando el software GIZA y se entrenan los pesos del modelo log-lineal con Mert, para finalmente, contrastar la salida con el conjunto de test y obtener el BLEU.

Como ya se había citado, no se han hecho pruebas sobre la variación del BLEU con el nº de iteraciones de MERT para entrenar los pesos del modelo log-lineal y todos los resultados que se muestran con Moses han alcanzado la convergencia.

Al igual que proponía el boletín, el modelo inicial será un modelo de trigramas con método de suavizado Kneser-Ney e interpolación con el que se consiguió un BLEU de 26.84.

2.1.1 Modelo sin ajuste de pesos

Tras la configuración base, la primera prueba que se realizó fue comprobar si fuese mejor el modelo de traducción sin ajuste de pesos, es decir, sin entrenar los pesos del modelo log-lineal con MERT, con lo que se obtuvo un BLEU de 26.37. Así pues, en adelante se trabajará con modelos con pesos ajustados ya que ofrecen mejores traducciones con este Corpus.

Modelo con pesos ajustados	**Modelo sin ajuste de pesos**
26.84	26.37

Tabla 1 – Valores de BLEU en función del ajuste de los pesos del modelo log-lineal

2.1.2 Variación valores n-gramas

Por lo que respecta a los valores de n-gramas se estudiará la variación del BLEU con más casos de lo que se hizo durante las prácticas para tratar de encontrar el mejor valor posible. De esta manera se han creado modelos desde 2-grama hasta 6-grama (recordando que 3-grama era el caso inicial) y una vez más, entrenando los pesos hasta convergencia con MERT.

Los resultados se pueden ver reflejados en la siguiente tabla, que, como se puede ver, los mejores resultados se obtuvieron con un modelo de 6-grama por lo que las siguientes pruebas se realizaron con un modelo de dicho tamaño de n-grama.

3-grama	**2-grama**	**4-grama**	**5-grama**	**6-grama**
26.84	25.92	26.69	26.89	26.91

Tabla 2 – Valores de BLEU en función del tamaño de n-grama

2.1.3 Evaluación MIRA

También se experimentó con MIRA para entrenar los pesos del modelo log-lineal en lugar de hacerlo con MERT, y aunque este algoritmo requiera de más iteraciones para converger, estas son más rápidas y al igual que sucedió con el Corpus EuTrans, en conjunto es más rápido que MERT. Por el contrario, a lo que sucedía en la práctica, con este corpus MIRA obtiene mejores prestaciones, tal como se ve reflejado a continuación, por lo que los pesos se siguieron entrenando con MIRA para conseguir el mejor modelo posible.

MERT	**MIRA**
26.91	26.94

Tabla 3 – Valores de BLEU en función del algoritmo de estimación de pesos

2.1.4 Técnicas de suavizado

Como ya se comprobó durante la práctica, un correcto suavizado puede llegar a tener gran importancia para obtener mejores resultados en traducción automática.

De nuevo, el tipo de suavizado que se empleó inicialmente es el de Kneser-Ney con interpolación, así que se probará a utilizar la técnica de backoff en lugar de interpolación tanto para el algoritmo de Kneser-Ney como Written-Bell. Finalmente se experimentará con suavizado de Laplace o 'añade uno' únicamente con backoff, ya que Moses no tiene implementada la versión con interpolación.

Como se puede ver en la siguiente tabla, los mejores resultados se obtuvieron con el suavizado que se empleó inicialmente, es decir, Kneser-Ney e interpolación.

	Interpolación	**Backoff**
Kneser-Ney	26.94	26.83
Written-Bell	26.14	26.38
Laplace	-	21.96

Tabla 4 – Valores de BLEU en función del método de suavizado

2.1.5 Moses monótono

Por último, se estimará el valor del BLEU sin reordenación en la traducción, es decir, empleando Moses monótono sobre el mejor modelo hasta el momento, es decir, 6-grama con suavizado Kneser-Ney e interpolación entrenado con MIRA.

El BLEU obtenido con Moses monótono ha sido de 27.00, que es ligeramente superior a la versión anterior y da el mejor resultado obtenido hasta ahora con cualquier modelo de traducción generado con Moses para esta tarea.

Moses dinámico	Moses monótono
26.94	27

Tabla 5 – Valores de BLEU en función de la capacidad de reordenación

2.1.6 Conclusiones

Con todas las pruebas que se han realizado en Moses, finalmente, el mejor modelo se ha obtenido con un modelo de 6-grama entrenado con MIRA hasta convergencia, empleando como técnica de suavizado Kneser-Ney con interpolación y Moses monótono (sin reordenación), con lo que se logró un BLEU de 27.

En conjunto, se ha observado en los resultados a través de todos los experimentos que, independientemente de la configuración a casi cualquier nivel, únicamente se producen ligeras variaciones en el BLEU que se obtiene tras la evaluación, es decir, generalmente, la configuración tiene poco impacto en el resultado final.

Es por ello por lo que es posible que los resultados obtenidos en las pruebas anteriores no sean fruto de una correcta configuración, sino, más bien que hayan surgido debido al componente aleatorio inherente a esta tarea.

En cualquier caso, se empleó el mejor modelo que se ha citado en párrafos anteriores con el conjunto de test que fue agregado posteriormente para tratar de obtener la mejor traducción posible.

2.2 Traducción con NMT-Keras

Al igual que con Moses, en el caso de NMT-Keras se partirá de una solución inicial y se tratará de ir mejorándola si es posible. Así pues, la configuración inicial será, de nuevo, la que proponía el boletín, es decir, codificador y decodificador LSTM de 64 neuronas, vector para codificar palabras fuente y destino (embeddings) de 64, factor de aprendizaje 0.001 con optimizador ADAM. Las pruebas se limitarán a 5 epochs y finalmente se entrenará de un modelo con la mejor configuración aumentando las epochs.

El procedimiento será idéntico al de las prácticas, entrenando el modelo de traducción siguiendo la configuración de config.py, para posteriormente realizar una traducción y evaluarla. Para el caso base con la configuración que se ha citado, se ha obtenido un BLEU de 17.95.

Cabe citar que en esta sección no se hará uso de Transformers con NMT-Keras, pues esa técnica se llevará a cabo en su respectivo capítulo de los ejercicios avanzados.

2.2.1 Variación de tamaño embedding

El primer experimento se realizó entorno a los tamaños de representación de las palabras fuentes y destino (word embeddings) donde la inicial era de 64. A partir de los resultados obtenidos, que se pueden ver en la siguiente tabla, el mejor tamaño de embedding para este Corpus ha sido de 128 con el que se ha logrado un BLEU de 19.73 con la configuración base, por lo que se mantuvo dicho tamaño para las siguientes pruebas.

Al igual que sucedía con el Corpus EuTrans, es posible que al incrementar el tamaño de embedding se reduzca el BLEU debido a que el conjunto de entrenamiento no es lo suficientemente grande (a pesar de ser mayor que el empleado en la tardea de EuTrans).

32	**64**	**128**	**256**	**512**
1.57	17.95	19.73	17.99	12.64

Tabla 6 – Valores de BLEU en función del tamaño de embedding

2.2.2 Variación tamaño red

A continuación, una vez definido el mejor tamaño de embedding, se tratará de estimar el mejor tamaño de la red, en otras palabras, la cantidad de neuronas que componen el decodificador y codificador LSTM. Una vez más, a continuación, se muestran las prestaciones que se han obtenido en función del número de neuronas de la red.

32	**64**	**128**	**256**	**512**
1.92	19.73	21.18	20.8	5.07

Tabla 7 – Valores de BLEU en función del tamaño de la red

A la vista de los resultados, se ha obtenido como mejor tamaño de la red 128 neuronas con un valor de BLEU de 21.18 mejorando el anterior, y sucede lo mismo que ocurría en el caso anterior, que, al incrementar esta cifra, empeora el BLEU debido a que no hay suficientes datos para que funcione correctamente.

2.2.3 Otros algoritmos de aprendizaje

Finalmente se probará a modificar el algoritmo optimizador o de aprendizaje partiendo de la mejor configuración encontrado hasta ahora, es decir, 128 de tamaño de embedding y 128 de tamaño de red con Adam como optimizador.

En la tabla que se muestra a continuación se reflejan los valores de BLEU que se han obtenido en función del optimizador que se ha empleado.

Adam	**Adagrad**	**Adadelta**	**SGD**
21.18	12.38	8.94	14.54

Tabla 8 – Valores de BLEU en función del algoritmo de aprendizaje

Tal como se puede ver, los mejores resultados que se han alcanzado han sido con Adam. Pese a esto, cabe destacar que, al igual que ocurrió en la práctica, no se encontró una configuración óptima que lograra un BLEU del mismo orden con el resto de los algoritmos probados, aunque puede deberse a la limitación de 5 epochs y que los resultados sean tan bajos debido a que su convergencia es mucho más lenta.

2.2.1 Conclusiones

En el caso de la experimentación llevada a cabo con el toolkit NMT-Keras, la configuración con la que se obtuvo el modelo que alcanzó un mayor valor de BLEU ha sido una topología LSTM (encoder-decoder con modelo de atención) con 128 tamaño de embedding, 128 tamaño de red y Adam como algoritmo de optimización, llegando a una cifra de 21.18 de BLEU.

Como se había comentado, tras las pruebas se aumentaría el número de epochs de entrenamiento para obtener el mejor modelo posible, pero, a pesar de haber incrementado este valor, no se han observado mejoras significativas en las prestaciones, siendo la variación del orden de décimas entre epochs superiores a cinco.

Por otro lado, y en contraposición a lo que ocurría en Moses, en este caso la configuración puede tener un gran impacto en el BLEU obtenido por el modelo generado, por lo que se ha tenido especial cuidado a la hora de la parametrización.

De igual manera a cómo ocurría en las prácticas, y en referencia a lo que ya se había citado, de manera generalizada los modelos basados en redes neuronales recurrentes (NMT-Keras) ofrecen peores prestaciones que los modelos estadísticos (Moses), probablemente debido a que los conjuntos de datos no son suficientemente grandes como para que las redes funcionan de forma óptima.

3. Ejercicios avanzados

3.1 Transformer NMT-Keras

En esta sección, se modificará la estructura de la red inicialmente propuesta, es decir, redes neuronales recurrentes basadas en encoder decoder con modelo de atención, por un Transformer.

Para ello se utilizarán el tamaño de embedding y algoritmo optimizador que dieron mejores resultados en los apartados anteriores (128 en ambos casos y Adam) mientras que se modificará el tamaño de la red para encontrar el mejor modelo de traducción basada en Transformer posible.

Una vez más, las pruebas que se han llevado a cabo han sido limitadas a 5 epochs, valor que se incrementará únicamente en el mejor de los casos, debido al coste computacional que conlleva, para maximizar el BLEU.

Los diferentes tamaños de modelo que se han probado se muestran en la tabla siguiente:

32	**64**	**128**	**256**	**512**
4.86	9.8	8.64	6	2.15

Tabla 9 – Valores de BLEU en función del tamaño del modelo

Como se puede ver, las opciones que se han valorado son las mismas con las que se experimentó tanto en el tamaño de embedding como en el tamaño de la red basada en encoder decoder con modelo de atención, y el mejor resultado que se ha obtenido ha sido con un tamaño de 128, que ha alcanzado un BLEU de 8.64.

A la vista de estos resultados, y al igual que sucedía en la práctica, la estructura de redes neuronales recurrentes basada en encoder decoder con modelo de atención ofrecía mejores resultados para esta tarea, ya que ninguno de los tamaños de Transformer propuestos alcanza los valores mostrados con anterioridad, debido a que un modelo de Transformer necesita aún más datos que uno basado en encoder decoder con modelo de atención.

3.2 OpenNMT

OpenNMT es un ecosistema de libre acceso para la traducción automática de tipo neuronal desarrollado por la universidad de Harvard. Actualmente su implementación está disponible a través de TensorFlow y PyTorch.

Para este proyecto se ha utilizado este toolkit sobre PyTorch en la misma computadora sobre la que se había ejecutado NMT-Keras (Windows con GPU NVIDIA GeForce GTX 1060 d 6GB) y se han probado algunas de las configuraciones propuestas en la propia documentación de OpenNMT o similares a las que se probaron con NMT-Keras.

La primera prueba que se hizo fue con la configuración base recomendada en la documentación de OpenNMT, esto es una LSTM con encoder bidireccional, adagrad como algoritmo optimizador, 128 tamaño de embedding y 512 de tamaño de red. Con esto se obtuvo un BLEU de 18.01, que es similar a los resultados obtenidos con NMT-Keras en algunas de sus configuraciones, pero que sigue sin igualar los resultados de los modelos estadísticos.

Además, también se probó a utilizar la topología de Transformer propuesta en la documentación, esto es, con 512 tamaño de embedding, 512 de tamaño de la red y Adam como algoritmo optimizador. Con esta arquitectura se obtuvo un valor de BLEU de 19.52, que, de nuevo, es inferior a los resultados de Moses, pero es considerablemente superior a los resultados obtenidos con Transformer con NMT-Keras.

Finalmente, se probó la configuración de Transformer que imita la de Google muy similar al Transformer anterior, modificando ligeramente algunos parámetros como el enfriamiento del factor de aprendizaje e incrementando el batch size. Con estas modificaciones se logró mejorar el BLEU hasta 19.65, pero no es una mejora significativa.

Los resultados se pueden ver resumidos en la siguiente tabla:

LSTM	**Transformer**	**Transformer 'Google'**
18.01	19.52	19.65

Tabla 10 – Valores de BLEU obtenidos con OpenNMT

4. Bibliografía

Casacuberta Francisco, & Domingo Miguel. (2021). *Boletín de la práctica*. Universitat Politècnica de València. https://www.prhlt.upv.es/~fcn/Students/ta/p1mt.pdf

Deniz, Y., & Andreas, S. (2007). *ngram-discount*. SRI International. http://www.speech.sri.com/projects/srilm/manpages/ngram-discount.7.html

European Union. (2018). *Moses - Main/HomePage*. LGPL. https://www.statmt.org/moses/

Harvard NLP Group. (2020). OpenNMT - Open-Source Neural Machine Translation. https://opennmt.net/