Traducción estadística basada en frases: Europarl

Jaime Ferrando Huertas

Enero 2021

Contents

1 Introducción

En este trabajo vamos a presentar todos los experimentos realizados para el trabajo final de la asignatura dedicado al corpus Europarl.

Hemos usado los conocimientos obtenidos en los dos primeros laboratorios para crear los mejores modelos SMT y NMT para nuestro nuevo corpus, Europarl. Se trata de un corpus europeo con traducciones en once idiomas distintos, aunque en nuestro caso solo hemos usado las traducciones ingles-español.

En el primer laboratorio hemos observado como el mejor modelo SMT estaba basado en un modelo de lenguaje de tri-gramas y reajuste de pesos con 10 iteraciones máximo, en este trabajo vamos a repetir dichos experimentos para encontrar los mejores parámetros para Europarl además de un análisis de procesamiento de datos de entrenamiento.

En el segundo laboratorio vimos como el mejor modelo RNN consistía de una red bastante pequeña y que la única modificación al experimento inicial que nos reportó mejores resultados fue aumentar el tamaño de los embeddings. En este trabajo queríamos evaluar modelos más grandes, pero debido a limitaciones de hardware y la imposibilidad de realizar experimentos exhaustivos hemos realizado una serie de experimentos a menor escala, estos han sido suficiente para evaluar como distintos parámetros de tamaño/entrenamiento afectan a nuestros modelos. Hemos realizado dos experimentos iniciales con distintos tamaños de la red: la mejor arquitectura del 2.º laboratorio y una arquitectura basada en la anterior pero de mayor tamaño. Con estos dos experimentos hemos sido capaces de ver que arquitecturas más grandes funcionan mejor para este nuevo corpus y hemos complementado la sección con 5 experimentos más donde tratamos otros aspectos de entrenamiento en modelos de gran calibre. Nos habría gustado hacer una evaluación más exhaustiva del tamaño de la red, epochs de entrenamiento, learning rate schedulers, pero el tiempo de entrenamiento en los escritorios virtuales ha sido el factor limitante. Todos los experimentos han sido comparados con la métrica BLEU [4].

También se han realizado los ejercicios opcionales, hemos creado un modelo de transformer con NMT-Keras y evaluado RNN/transformers en el toolkit de OpenNMT.

Para los ejercicios básicos hemos usado las máquinas proporcionadas por el DSIC mientras que para openNMT hemos podido ejecutar en Google Colab.

2 Datos

Europarl [3] es un corpus multilenguage para traducción automática, cuenta con transcripciones del parlamento europeo y es disponible públicamente. Nosotros hemos usado el subconjunto de ingles-español para nuestros modelos.

El corpus de Europarl no cuenta con conjunto de development por lo que hemos decidido el conjunto de train en 2 partes, 45.000 frases train y 5.000 development. Este segmento de development era necesario para el ajuste de pesos log-lineal de nuestro modelo SMT y para el proceso de evaluación en

modelos de redes. También se ha usado las herramientas de MOSES para limpiar y tokenizar nuestros datos de entrenamiento.

3 Ejercicios básicos

3.1 SMT

Para crear el mejor modelo de SMT vamos a repetir los mismos experimentos del laboratorio primero con nuestro dataset de Europarl. Queremos modificar el número máximo de iteraciones de MERT y probar distintos N-gramas en el modelo de lenguaje, lo correcto seria realizar experimentos combinatorios de estos dos parámetros, pero dado el coste computacional se han realizado por separado y posteriormente creado un modelo con el mejor valor observado para cada parámetro. Una vez encontrado los mejores parámetros para el número de iteraciones MERT y n-gramas hemos probado 3 variantes del modelo resultante, cada una con distintos métodos de suavizado.

Experimento	MERT Iter	N-Gramas	Suavizado	Monótono	BLEU
N-GRAMAS 2	5	2	Kneser-Ney	NO	25.06
N-GRAMAS 3	5	3	Kneser-Ney	NO	**25.30**
N-GRAMAS 4	5	4	Kneser-Ney	NO	25.04
N-GRAMAS 5	5	5	Kneser-Ney	NO	25.05

Table 1: Europarl: Evolución del BLEU dado n-gramas del modelo de lenguaje.

Vemos como el modelo de lenguaje con tri-gramas se mantiene como mejor opción para nuestro nuevo corpus. Vemos también que la diferencia en BLEU entre distintos modelos de lenguaje no es tan alta como en EuroTrans, pero pensamos que es debido a que el BLEU es muy bajo y por tanto es una diferencia relativa. Procedemos ahora a experimentar con distintas iteraciones de MERT junto al modelo de lenguaje basado en tri-gramas.

Experimento	MERT Iter	N-Gramas	Suavizado	Monótono	BLEU
MERT 5	5	2	Kneser-Ney	NO	25.30
MERT 7	5	3	Kneser-Ney	NO	25.19
MERT 10	5	4	Kneser-Ney	NO	**25.44**
MERT 15	5	5	Kneser-Ney	NO	25.20

Table 2: Europarl: Evolución del BLEU dadas iteraciones de MERT en ajuste de pesos.

Encontramos ahora que diez iteraciones es el valor óptimo Europarl al contrario del valor cinco que vimos para EuroTrans. Puede que esto sea debido a que tenemos un conjunto de desarrollo (usado en el ajuste de pesos) frente a los experimentos de EuroTrans. Una vez visto el valor óptimo de n-gramas e iteraciones de MERT queremos probar 3 métodos de suavizado.

Experimento	MERT Iter	N-Gramas	Suavizado	Monótono	BLEU
Witten-Bell	10	3	Witten-Bell	NO	25.02
Kneser-Ney	10	3	Kneser-Ney	NO	**25.44**
Good-turing	10	3	Good-turing	NO	25.11

Table 3: Europarl: Evolución del BLEU dado método de suavizado.

A diferencia de nuestros experimentos con EuroTrans (donde Witten-Bell era el mejor método) Kneser-ney es la mejor opción para Europarl.

Una vez encontrado el mejor conjunto de parámetros de entrenamiento para nuestro modelo SMT hemos querido añadir un análisis extra frente al primer laboratorio. En este análisis vamos a comparar como el preprocesado de datos hecho en MOSES puede afectar a nuestro modelo. Hemos evaluado como la restricción de longitud de frase aplicada al limpiar el corpus con $clean-corpus-n.perl$ afecta al BLEU entrenando nuestro mejor modelo hasta la fecha (25.44 BLEU) con los distintos sets de entrenamiento obtenidos. Hemos probado dos longitudes de frases nuevas frente a la predeterminada del boletín de practicas (60).

Experimento	MERT Iter	N-Gramas	Suavizado	Monótono	BLEU
clean 1 60	10	3	Kneser-Ney	NO	25.44
clean 1 80	10	3	Kneser-Ney	NO	**27.33**
clean 1 100	10	3	Kneser-Ney	NO	26.21

Table 4: Europarl: Evolución del BLEU dado longitud de procesado en MOSES.

Vemos que la longitud de frase 80 nos reporta un gran incremento en BLEU (casi 2 puntos) pero que aumentarla a 100 no reporta tanta mejoría aun siendo mejor que longitud 60. A la hora de entrenar nuestros modelos de traducción siempre queremos mas datos pero al mismo tiempo queremos que estos datos estén limpios para que el modelo sea capaz de aprender lo que nos interesa. En este caso al procesar los datos con longitud 80 estamos dejando de eliminar tantas frase como con longitud 60 pero potencialmente introduciendo frases mas complejas que puedan perjudicar al modelo. Para nuestro caso la longitud 80 es el punto óptimo en el balance mencionado. Pensamos que longitud 100 no reporta mejores resultados ya que la ganancia en número de muestras es mucho menor frente a 60 vs 80 pero acepta muestras de entrenamiento que penalizan a nuestro modelo.

Tenemos entonces ahora el mejor modelo para Europarl (procesado 1-80, trigramas, MERT 10, suavizado de Kneser-ney) el cual ha sido usado para evaluar el test oculto proporcionado.

3.2 NMT-Keras

Para evaluar si la creación de modelos neuronales de mayor tamaño era beneficioso para nuestro caso hemos realizado dos experimentos iniciales, uno con la

mejor arquitectura del segundo laboratorio y otro con una arquitectura de mayor tamaño. De esta manera evaluaremos rápidamente si es viable o no continuar con experimentos de este calibre. Se ha repetido el experimento de mejores resultados del segundo laboratorio y hemos añadido una versión modificada del primero "grande" con mayor tamaño de embeddings a entrada y salida además de mayor número de neuronas. Los dos modelos han sido entrenados durante 5 epochs siguiendo las direcciones del correo enviado por Paco.

Experimento	Emb. Size	Enc. Size	Dec. Size	Optimizer-lr-Epochs	BLEU
Modelo 2do laboratorio	128	1x64	1x64	Adam-0.001-5	11.26
Modelo grande	2048	1x512	1x512	Adam-0.001-5	**22.62**

Table 5: Experimentos iniciales NMT-Keras, Europarl

Podemos ver que nuestro modelo de gran tamaño obtiene resultados mucho mejores y se acerca a los obtenidos por nuestros experimentos de SMT. Pensamos que estos experimentos dejan claro el beneficio de redes de mayor tamaño para problemas más complejos como este nuevo corpus y debido a las limitaciones de entrenamiento en las máquinas del DSIC no vamos a evaluar distintos parametros para optimizar al máximo el tamaño. Ahora nos gustaría evaluar otros dos parámetros (epochs y learning rate) sobre este modelo grande para evaluar si con ellos conseguiremos mejores resultados. Nos disponemos ahora a realizar cuatro variantes del experimento modelo grande donde exploraremos mayor número de epochs y menor learning rate.

Experimento	Emb. Size	Enc. Size	Dec. Size	Optimizer-lr-Epochs	BLEU
Grande-1	2048	1x512	1x512	Adam-0.001-5	22.62
Grande-2	2048	1x512	1x512	Adam-0.001-15	20.73
Grande-3	2048	1x512	1x512	Adam-0.0005-5	20.34
Grande-4	2048	1x512	1x512	Adam-0.0005-15	**23.05**

Table 6: Experimentos con distinto número de epochs y learning rate, NMT-Keras Europarl

Podemos observar que aumentar el número de epochs sin reducir el learning rate nos lleva a una reducción de BLEU (experimento Grande-1 vs Grande-2, overfitting). Que reducir el learning rate pero mantener un mismo número de epochs también conlleva una reducción del BLEU (experimento Grande-3) pero que si reducimos el learning rate y aumentamos el número de epochs podemos ver mejoras en el BLEU (experimento Grande-4). Estos resultados no son sorpresa para nosotros, ya que es normal que a un learning rate menor el optimizador necesite mas epochs para reducir el valor de nuestra función de perdida y también sea capaz de encontrar mínimos locales más bajos. Podríamos realizar una analogía del learning rate como el tamaño de pisadas que realizamos al caminar por el espacio de nuestra función de perdida, si nuestras pisadas son pequeñas podremos entrar en espacios más pequeños (valores más bajos en la

función de perdida) pero también necesitaremos mayor número de pisadas para recorrer una distancia (mayor número de epochs).

Para la entrega del test oculto vamos a restringir el número de epochs de entrenamiento a 5 (siguiendo el correo de Paco) por lo que también queremos realizar un último experimento con un learning rate mayor para ver si conseguimos acércanos a los resultados del experimento Grande-4 en un número menor de epochs.

Experimento	Emb. Size	Enc. Size	Dec. Size	Optimizer-lr-Epochs	BLEU
Grande-5	2048	1x512	1x512	Adam-0.0025-5	**24.65**

Table 7: Experimento para predicciones test oculto, NMT-Keras Europarl

Encontramos el mejor resultado hasta la fecha y sigue en las líneas de lo comentando previamente, para número de epochs bajo es recomendable learning rates bajos mientras que si aumentamos el número de epochs lo querremos reducir para poder encontrar mejores mínimos locales. Este ha sido el modelo usado para realizar las evaluaciones del test oculto.

4 Ejercicios extras

4.1 Transformer NMT-Keras

Para el primer ejercicio extra hemos probado a ejecutar la arquitectura de transformer [5] y hemos repetido los mismos experimentos que en el laboratorio. Esta arquitectura itera sobre los modelos encoder-decoder y sustituye las capas de LSTM por capas de atención para guardar el contexto temporal. Para nuestro experimento hemos escogido la arquitectura que nos reporto mejor BLEU en el laboratorio de NMT-Keras y ejecutado con los mismos parámetros (también el número de epochs y learning rate). Los resultados obtenidos:

Experimento	Emb. Size	Enc. Size	Dec. Size	Optimizer-lr-Epochs	BLEU
Transformer-64	64	1x64	1x64	Adam-0.001-5	3.55
Transformer-128	128	1x64	1x64	Adam-0.001-5	3.03
Transformer-256	256	1x64	1x64	Adam-0.001-5	**3.71**

Table 8: Evolución del BLEU para distintos tamaños de transformer

Podemos ver que los resultados son totalmente catastróficos, no hay ningún indicio del porqué 256 deba ser el mejor tamaño de transformer para este corpus y creemos que la razón de que obtenga el mejor bleu no es nada más que una mejor inicialización de pesos aleatoria. La arquitectura de transformers es una arquitectura bastante compleja y que requiere de un proceso largo (muchas epochs) y lento (bajo learning rate) para conseguir sacar beneficio de ella. Vamos a realizar un experimento ahora con 20 epochs para evaluar brevemente la posibilidad de entrenamiento de estos modelos con mas epochs.

Experimento	Emb. Size	Enc. Size	Dec. Size	Optimizer-lr-Epochs	BLEU
Transformer-256	256	1x64	1x64	Adam-0.001-20	**13.21**

Table 9: Transformer 20 epochs

Vemos como el aumentar el número de epochs nos ha reportado mejor bleu aunque no nos acercamos a los resultados obtenidos anteriormente con los mejores modelos SMT/NMT. Aconsejaríamos realizar un estudio similar al hecho en el apartado de NMT-Keras para poder evaluar que parámetros usar en el entrenamiento de Transformers.

4.2 Modelos de traducción con openNMT

Para este último experimento hemos usado el toolkit openNMT[2] (en su version de pytorch) para crear modelos de traducción basados en deep learning. Es un toolkit del grupo de NLP en Harvard y nos permite trabajar con redes neuronales de una manera muy cómoda, pudiendo ejecutarse también sobre gpu. Para hacer uso del mismo hemos usado un tutorial para ejecutarlo sobre notebooks en Google Colab de manera que teníamos acceso a gpus de gran capacidad de cómputo. Hemos probado sus dos configuraciones predeterminadas para modelos Transformers y RNN encoder-decoder (basado en LSTM [1] como el modelo de NMT-Keras), gracias a ejecutar en Google Colab y sus gpus estos modelos han podido ejecutar las epochs necesarias para llegar al punto de entrenamiento óptimo. En este punto es donde las métricas sobre el conjunto de validación empeoran y empezamos a crear lo que se llama "overfit" sobre el conjunto de entrenamiento.

Experimento	Emb. Size	Enc. Size	Dec. Size	Optimizer-lr-Epochs	BLEU
openNMT-RNN	32	1x64	1x64	Adam-0.0001-33	19.88
openNMT-Trans	512	1x64	1x64	Adam-0.0001-45	24.07

Table 10: Evolución del BLEU dado tamaño de embeddings de entrada y salida

Podemos ver que el modelo básico de RNN no se acerca a los resultados vistos en NMT-Keras con los experimentos Grande-5 y Grande-4 pero sin embargo consigue mejores resultados que su experimento análogo(Modelo 2do laboratorio) gracias a ese mayor número de epochs de entrenamiento. Vemos también como el experimento con transformers consigue resultados completamente distintos a los obtenidos en nuestros experimentos con NMT-Keras y pensamos que esto se debe a las 45 epochs de entrenamiento junto a un conjunto de parámetros optimizados proporcionados por la configuracion de openNMT.

Para concluir pensamos que openNMT es un toolkit sencillo de usar y que puede ser usado para fácil acceso a entrenamiento de modelos de traducción automática en gpu gracias a sus notebooks alojados en Google Colab.

5 Conclusión

Con este trabajo hemos podido entrar en contacto con un corpus más "sucio" y difícil de aprender para nuestros modelos que el EuroTrans visto en los laboratorios. Nuestro modelo de MOSES conseguía mejores puntuaciones que nuestros otros modelos basados en redes(27.33 vs 24.65). Hemos visto el problema de los modelos neuronales y su complejidad que los hacen difícilmente entrenables sin el hardware adecuado, nuestros experimentos nos dicen que mayores modelos y entrenados durante mas tiempo y con menor learning rate son mas óptimos.. Es cierto que el estado del arte de traducción automática esta generalmente dominado por modelos neuronales pero pensamos que aún hay cabida para modelos estadísticos en situaciones como esta en la que no se cuenta con gpus.

Para trabajos futuros nos gustaria realizar mas experimentos en la sección de NMT-Keras tanto con distintos optimizadores como número de capas. También queremos explorar el procesado de datos y como estos afectan a nuestro mejor modelo NMT de misma forma que hemos hecho con MOSES.

Nos gustaría sugerir que para próximos cursos se elaborara un notebook de Google Colab a modo de tutorial y base para el toolkit de NMT-keras, de esta forma se conseguiría que los estudiantes tengan acceso a gpus y prueben modelos mas complejos. En el toolkit de openNMT encontramos uno de estos notebooks y nos ha sido de gran ayuda. Se pueden encontrar ahora tablas con todos los experimentos realizados.

Experimento	MERT Iter	N-Gramas	Suavizado	Monótono	BLEU
N-GRAMAS 2	5	2	Kneser-Ney	NO	25.06
N-GRAMAS 3	5	3	Kneser-Ney	NO	25.30
N-GRAMAS 4	5	4	Kneser-Ney	NO	25.04
N-GRAMAS 5	5	5	Kneser-Ney	NO	25.05
MERT 5	5	2	Kneser-Ney	NO	25.30
MERT 7	5	3	Kneser-Ney	NO	25.19
MERT 10	5	4	Kneser-Ney	NO	25.44
MERT 15	5	5	Kneser-Ney	NO	25.20
Witten-Bell	10	3	Witten-Bell	NO	25.02
Kneser-Ney	10	3	Kneser-Ney	NO	25.44
Good-turing	10	3	Good-turing	NO	25.11
clean 1 60	10	3	Kneser-Ney	NO	25.44
clean 1 80	10	3	Kneser-Ney	NO	**27.33**
clean 1 100	10	3	Kneser-Ney	NO	26.21

Table 11: Experimentos realizados para SMT con MOSES, Europarl

Experimento	Emb. Size	Enc. Size	Dec. Size	Optimizer-lr-Epochs	BLEU
Modelo 2do laboratorio	128	1x64	1x64	Adam-0.001-5	11.26
Modelo grande	2048	1x512	1x512	Adam-0.001-5	22.62
Grande-1	2048	1x512	1x512	Adam-0.001-5	22.62
Grande-2	2048	1x512	1x512	Adam-0.001-15	20.73
Grande-3	2048	1x512	1x512	Adam-0.0005-5	20.34
Grande-4	2048	1x512	1x512	Adam-0.0005-15	23.05
Grande-5	2048	1x512	1x512	Adam-0.0025-5	**24.65**
Transformer-64	64	1x64	1x64	Adam-0.001-5	3.55
Transformer-128	128	1x64	1x64	Adam-0.001-5	3.03
Transformer-256	256	1x64	1x64	Adam-0.001-5	3.71
Transformer-256	256	1x64	1x64	Adam-0.001-20	13.21
openNMT-RNN	32	1x64	1x64	Adam-0.0001-33	19.88
openNMT-Trans	512	1x64	1x64	Adam-0.0001-45	24.07

Table 12: Experimentos realizados para NMT, Europarl

References

[1] Sepp Hochreiter and Jürgen Schmidhuber. Long short-term memory. *Neural computation*, 9:1735–80, 12 1997.

[2] Guillaume Klein, Yoon Kim, Yuntian Deng, Jean Senellart, and Alexander Rush. OpenNMT: Open-source toolkit for neural machine translation. In *Proceedings of ACL 2017, System Demonstrations*, pages 67–72, Vancouver, Canada, July 2017. Association for Computational Linguistics.

[3] Philipp Koehn. Europarl: A parallel corpus for statistical machine translation. 5, 11 2004.

[4] 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 of the Association for Computational Linguistics*, pages 311–318, Philadelphia, Pennsylvania, USA, July 2002. Association for Computational Linguistics.

[5] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N. Gomez, Lukasz Kaiser, and Illia Polosukhin. Attention is all you need, 2017.