Trabajo final de Traducción Automatica

Marcos Esteve Casademunt

Febrero 2020

Contents

1 Introducción

2 Objetivos

El objetivo principal de este trabajo consiste en traducir los tweets de la tarea MultiStanceCat[1] del catalán y el español al inglés para posteriormente utilizar un software[2] de detección de argumentos desarrollado por Iryna Gurevyc en la Universidad Técnica de Darmstadt.

3 Datasets

Para la realización del trabajo es necesario disponer de corpus paralelos para entrenar los traductores automáticos. En nuestro caso será necesario un traductor del catalán al español y otro del español al inglés que se encarguen de traducir los *tweets* del corpus MultiStanceCat al inglés para que, de esta forma, puedan ser utilizados por el software desarrollado para la extracción de argumentos.

3.1 Catalán-Español

Para la construcción del traductor del catalán al español se ha utilizado el corpus paralelo DOGC[3]. Se trata de un conjunto de 4,8 Millones de pares del diario oficial de la Generalitat de Catalunya en catalán y castellano aportados por Antoni Oliver González de la Universitat Oberta de Catalunya. Destacar que el dataset se proporciona en un fichero TMX el cual deberá ser limpiado y preprocesado para poder ser utilizado.

3.2 Español-Inglés

En cuanto a la construcción del traductor del español al inglés se ha optado por utilizar el corpus paralelo de Europarl, el cual se extrae de las actas del Parlamento Europeo. Este corpus incluye versiones en 21 lenguas europeas. Destacar que el corpus que nos interesa consta de un total de 1,9 Millones de pares de frases español-ingles. El corpus no viene limpio ni tokenizado, por lo que será necesario realizar este proceso antes de entrenar el modelo de traducción.

3.3 MultiStanceCat

Se trata de un dataset formado por 5545 *tweets* en español y 5853 *tweets* en catalán. Ejemplos de estos tweets los podemos observar en la figura 1

[1] https://www.autoritas.net/MultiStanceCat-IberEval2018/
[2] https://www.argumentext.de/
[3] http://opus.nlpl.eu/DOGC.php

- @emaspons @ABecharaGhobril @UnitsCat @Espinakka @rccarbo Publicada fa dies, abans que el posicionament #1 oct, aquí... K2
- Marianicooo. Que no te enfrentas a políticos! Te enfrentas a todo un pueblo! # JaEmGuanyat # 1O https://t.co/tNRSEc3ZhF
- Señor @marianorajoy, porque ha cambiado usted de opinión? https://t.co/OODB4aOJay # Votarem # catalunyanouestatdeuropa # 1Oct2017

Figure 1: Ejemplos de *tweets* del corpus MultiStanceCat

Como podemos observar los *tweets* tienen un lenguaje más espontaneo además de utilizar ciertas características propias de esta red social como los *hashtags*, las menciones y los links. Estas características no aportan ninguna información necesaria y más bien introducen ruido en el sistema, por lo que será necesario realizar un preproceso previo que nos permita eliminar estas características.

4 Propuesta de solución

Dado que el corpus a traducir se trata de un corpus de lenguaje coloquial, espontaneo y los corpus paralelos que se disponen se tratan de textos formales, se propone un sistema basado en redes neuronales y que mediante una post-edición permita al operario experto traducir los *tweets* utilizando el mínimo esfuerzo posible.

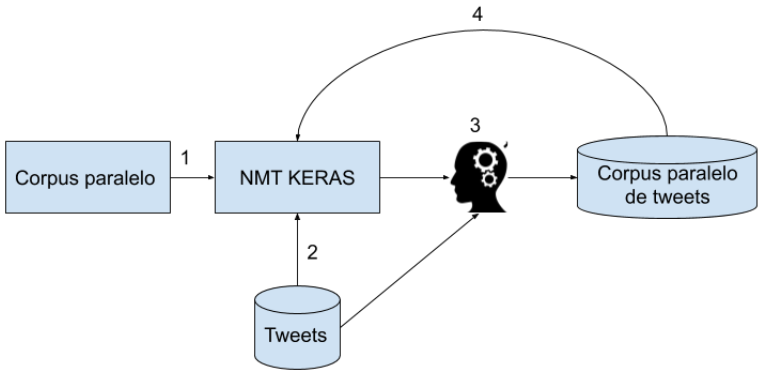

Figure 2: Sistema propuesto para traducir los *tweets*

Tal y como se puede observar en la Figura 2 el sistema consta de varias partes.

En primer lugar (1) mediante uno de los dos corpus paralelos expuestos en 3.1 o 3.2 se entrenan los traductores basados en redes neuronales con el toolkit NMT-keras[4]. Una vez entrenado el modelo de traducción en (2) se utiliza un subconjunto del corpus de *tweets* expuesto en 3.3 y se traduce con el modelo de traducción. Tras esto, en (3) se le pasa al usuario experto el *tweet* original y

[4]https://github.com/lvapeab/nmt-keras

el *tweet* traducido por el sistema de traducción, este usuario deberá de realizar un proceso de post-edición corrigiendo los errores de traducción del tweet y elaborando de esta manera un corpus paralelo de tweets que podrá ser utilizado para realizar un ajuste de los pesos del modelo de traducción neuronal en (4).

Siguiendo el esquema expuesto en la figura 2 se pueden generar los modelos de traducción catalán-español y español-ingles. Una vez entrenados estos modelos se puede definir un *pipeline* de traducción como el descrito en la siguiente figura.

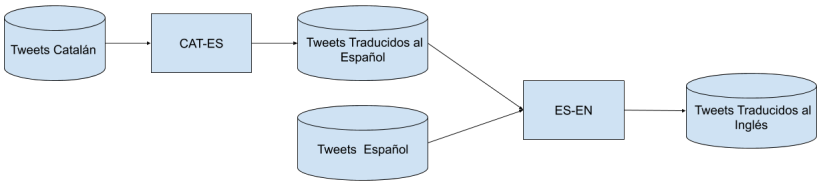

Figure 3: *Pipeline* de traducción de *tweets* de catalán y español al inglés

Tal como podemos observar en la figura 3, en primer lugar, se traducirán los *tweets* del catalán al español para posteriormente utilizar el traductor español-inglés para obtener la traducción final en este idioma.

5 Preproceso *Tweets*

Uno de los principales problemas que poseen los *tweets*, tal y como hemos comentado en el punto 3.3, es la cantidad de variabilidad y espontaneidad que expresan los usuarios. A continuación se muestran algunos de las expresiones que realizan los usuarios en esta red social:

- uso de *hashtags*
- uso menciones a otros usuarios
- uso de una gran cantidad de abreviaciones propias del lenguaje coloquial como por ejemplo "q" para referirse a "que" o "xq" para referirse a "por qué".
- uso de enlaces a webs
- incremento del numero de vocales o consonantes en una palabra como por ejemplo "Marianicoooo"

El uso de estas expresiones supone un problema a la hora de ser traducidas ya que los corpus paralelos que disponemos están basados en textos formales, los cuales no utilizan ninguna de estas expresiones, y por tanto, los traductores entrenados no obtendrán buenos resultados al enfrentarse a estas características del lenguaje coloquial.

En un intento por paliar esta situación se ha desarrollado un programa en Python que permite eliminar gran parte de los comportamientos enumerados anteriormente

6 Construcción del traductor Catalán-Español

En esta sección se comentaran los distintos pasos seguidos para la construcción del traductor del catalán al español. Desde la división del conjunto de datos al estudio de la mejor arquitectura neuronal para resolver el problema.

6.1 División del conjunto de datos

Debido a las limitaciones en el hardware de la máquina donde se han ejecutado los experimentos se ha optado por escoger un subconjunto de 180.000 muestras para entrenamiento, 2500 muestras para test y 1000 muestras para desarrollo.

6.2 Construcción del sistema

A la hora de determinar la mejor arquitectura neuronal se ha realizado una exploración sobre el modelo de traducción basado en la arquitectura encoder-decoder con atención y el modelo basado en transformers

6.2.1 Encoder-decoder

A la hora de evaluar la arquitectura de encoder-decoder se ha realizado en primer lugar una exploración exhaustiva por el tamaño del *embedding* y posteriormente una vez fijado el tamaño del *embedding* se ha variado el tamaño de la red LSTM tanto en el input como en el output. A continuación se muestra el BLEU obtenido en test al variar el tamaño del *embedding* fijando el tamaño de las redes LSTM a 32 unidades

Table 1: Evolución del BLEU al modificar el tamaño de los *embeddings* (1 epochs)

Input Embedding	**Target Embedding**	**BLEU**
32	32	47.94
64	64	57.86
128	**128**	**66.05**
256	256	60.71

A la vista de los resultados expuestos en la tabla 1 se observa que el mejor resultado se obtiene con tamaño de *embeddings* del input y del target 128. Una vez fijado el tamaño de embedding a 128 pasamos a analizar cual es la contribución al BLEU de los distintos tamaños de la red LSTM

Table 2: Evolución del BLEU al modificar el tamaño de las capas LSTM (1 epochs)

Input LSTM	**Target LSTM**	**BLEU**
32	32	66.05
64	**64**	**68.69**
128	128	66.31

A la vista de los resultados se observa que el mejor modelo obtenido para la traducción catalán-español se obtiene con un tamaño de embedding para el

input y el target de **128** y un tamaño para las redes LSTM de **64**. Una vez fijado los tamaños óptimos se ha realizado un entrenamiento de 5 epochs con el cual se ha obtenido un BLEU de **74.12**

6.2.2 Transformer

Para evaluar el modelo basado en transformers se han explorado distintos tamaños del modelo. A continuación se muestran los resultados obtenidos.

Table 3: Evolución del BLEU al variar el tamaño del modelo Transformer (1 epochs)

Tamaño del modelo	**BLEU**
32	11.42
64	39.34
128	**52.78**
256	29.23

A la vista de los resultados expuestos en la tabla 3 se observa que el modelo basado una arquitectura de transformers y con un tamaño de 128 es el que consigue el BLEU más alto. Cabe destacar que el BLEU del modelo basado en transformers se situa por debajo del mejor modelo obtenido con encoder-decoder.

7 Construcción del traductor Español-Inglés

En esta sección se seguirá la misma estrategia que en el punto anterior, explicando los distintos pasos seguidos para la construcción del traductor del español al inglés.

7.1 División del conjunto de datos

Debido a las limitaciones en el hardware de la máquina se ha seguido la misma estrategia que en el punto anterior y se ha optado por escoger 180.000 muestras para entrenamiento, 2500 muestras para test y 1000 muestras para desarrollo.

7.2 Construcción del sistema

De nuevo, se ha seguido la misma estrategia que en el modelo anterior. Se ha realizado una exploración de los dos modelos que nos proporciona nmtkeras con el fin de determinar el mejor traductor neuronal. A continuación se muestran los experimentos realizados utilizando la arquitectura Encoder-Decoder y los Transformers. Cabe destacar que construir un traductor del Español al inglés es más complicado que del Catalán al Español debido a las diferencias que existen a nivel sintáctico entre las lenguas. Por esta razón ha sido necesario un mayor número de *epochs* para que el modelo obtuviera buenos resultados.

7.2.1 Encoder-decoder

Se ha seguido la misma estrategia que en el entrenamiento del modelo de traducción catalán-español. Primeramente se han explorado distintos tamaños de *embeddding* fijando para ello el tamaño de la red LSTM a 64 unidades y posteriormente se ha explorado la influencia del tamaño de la red.

Table 4: Evolución del BLEU al modificar el tamaño de los embeddings (3 epochs)

Input Embedding	**Target Embedding**	**BLEU**
64	64	22.81
128	**128**	**30.99**
256	256	30.01

A la vista de los resultados expuestos en la tabla 4 se observa como el traductor con tamaño de *embeddings* 128 es el que mejor BLEU ha manifestado. A continuación se muestra la evolución del BLEU al modificar el tamaño de la red LSTM y fijando el tamaño de *embeddings* a 128 unidades.

Table 5: Evolución del BLEU al modificar el tamaño de las capas LSTM (3 epochs)

Input LSTM	**Target LSTM**	**BLEU**
64	64	30.99
128	**128**	**32.12**
256	256	31.43

Como podemos ver en los resultados expuestos en la tabla 5 se observa que el mejor modelo obtenido utilizando la arquitectura Encoder-Decoder se obtiene utilizando tamaños de *embedding* de 128 dimensiones y 128 unidades LSTM. Una vez establecido la mejor combinación de los parámetros se ha realizado un entrenamiento de 6 *epochs* consiguiendo un BLEU de **33.32**.

7.2.2 Transformer

Table 6: Evolución del BLEU al variar el tamaño del modelo Transformer (3 epochs)

Tamaño del modelo	**BLEU**
64	23.08
128	**26.64**
256	15.20

A la vista de los resultados expuestos en la tabla 6 se observa que los mejores resultados se obtienen con un tamaño de modelo de 128 unidades. Destacar que el modelo basado en la arquitectura de Transformers consigue unos resultados significativamente inferiores al mejor modelo obtenido con la arquitectura encoder-decoder

8 Traducción y post-edición de los *Tweets*

Una vez entrenados los modelos de traducción para el catalán-español y español-inglés se ha procedido a traducir los *tweets* sin realizar un proceso de ajuste de los modelos. Tal y como se pueden observar en los ficheros anexos **esTweets0PosEdit** y **enTweets0PosEdit** los resultados no tenían ninguna calidad.

Con el objetivo de mejorar esta situación se ha utilizado, tal y como se comenta en la sección 4, los modelos como una asistencia a la traducción.

Para realizar este proceso con el toolkit nmt-keras ha sido necesario cargar una *epoch* previamente entrenada con uno de los corpus formales para posteriormente, traducir un subconjunto de los *tweets*, corregirlos y re-alimentarlos a la red para ajustar los modelos. A continuación se comentan algunos aspectos interesantes que se han podido observar en la post-edición y mejora ambos modelos de traducción.

8.1 Catalán - Español

A la hora de traducir los *tweets* del catalán al español se ha visto que los *tweets* tienen una gran cantidad de faltas de ortografía, lo que hace la tarea de traducción más complicada.

Tal y como se puede observar en el fichero anexo **esTweets0PosEdit** los resultados iniciales sin el proceso de post-edición no poseen ninguna calidad. Observando los resultados de los ficheros anexos **esTweets100PosEdit**, **esTweets500PosEdit** y **esTweets1000PosEdit** se observa que aunque se mejora la calidad de los resultados obtenidos al incrementar el número de muestras post-editadas, el sistema sigue mostrando numerosos errores.

Finalmente, en el fichero **esTweetsTotalPosEdit** se pueden observar los resultados tras 2500 muestras post-editadas, donde tal y como se puede observar, siguen existiendo una gran cantidad de errores.

8.2 Español - Inglés

Por otra parte, a la hora de traducir los *tweets* del español al inglés si que se ha observado una mejora sustancial al incrementar el número de muestras post-editadas para el ajuste del modelo.

Tal y como se puede observar en el fichero anexo **enTweets0PosEdit** los resultados eran escasos y era necesario realizar múltiples correcciones. Al ajustar el modelo incrementando el número de muestras post-editadas los resultados iban mejorando tal y como se puede observar en los ficheros anexos **enTweets100PosEdit**, **enTweets500PosEdit** y **enTweets1000PosEdit**.

Por último, en el fichero **enTweetsTotalPosEdit** se pueden observar los resultados tras 3000 muestras post-editadas, donde tal y como se puede observar, los resultados han mejorado significativamente aunque, siguen existiendo algunos errores.

9 Argument mining

La minería de argumentos es un área de investigación dentro del campo del procesamiento de lenguaje natural cuyo objetivo es la extracción e identificación

automática de estructuras argumentativas a partir de un texto.

Una vez traducidos los *tweets* del catalán y el español al inglés se ha procedido ha emplear la herramienta de minería de argumentos desarrollada por la Universidad Técnica de Darmstadt. Para ello es necesario especificar un tópico por lo que se ha decidido utilizar los tópicos **polarization** y **stance**. Notar que los tweets cuyo nombre de fichero es [CONTRA/FAVOR/NEUTRAL]CAT Son los tweets traducidos del catalán al ingles con un posicionamiento [CONTRA/-FAVOR/NEUTRAL] en el dataset MultiStanceCat. Lo mismo ocurre con los ficheros [CONTRA/FAVOR/NEUTRAL]ES donde la traducción es del español al inglés

Table 7: Análisis de argumentos para el tópico *polarization*

Fichero	Args	Oraciones	ContraArgs	NoArgs	ProArgs
CONTRACAT	2	116	2	114	0
CONTRAES	118	1702	81	1584	37
FAVORCAT	184	3578	89	3394	95
FAVORES	68	1632	45	1564	23
NEUTRALCAT	27	454	15	427	12
NEUTRALES	43	876	27	833	16

Table 8: Analisis de argumentos para el tópico *stance*

Fichero	Args	Oraciones	ContraArgs	NoArgs	ProArgs
CONTRACAT	2	116	2	114	0
CONTRAES	112	1702	81	1590	31
FAVORCAT	180	3578	98	3398	82
FAVORES	74	1632	51	1558	23
NEUTRALCAT	22	454	14	432	8
NEUTRALES	46	876	31	830	15

A la vista de los resultados expuestos en las tablas superiores se observa que el porcentaje de argumentación existente en los tweets es muy bajo en todas las comunidades existentes. En los ficheros adjuntos **argumentos_polarization** y **argumentos_stance** se pueden ver los argumentos extraidos por la herramienta así como el posicionamiento que les asigna.

10 Conclusiones

A la vista de los resultados expuestos en los apartados anteriores podemos extraer las siguientes conclusiones.

En primer lugar, tal y como se ha comentado en las clases de teoría. Si tenemos un modelo entrenado para un dominio concreto, por ejemplo un dominio legal, y tratamos de aplicarlo por ejemplo a un dominio médico, en primera instancia los resultados no serán adecuados y por tanto será necesario un proceso de ajuste del modelo basado en leyes para que traduzca adecuadamente el dominio médico. Siguiendo esa misma aproximación se ha intentado adaptar los traductores entrenados sobre un dominio formal a un dominio informal como

puede ser Twitter. Uno de los principales problemas que se ha encontrado en esta adaptación es la propia jerga informal utilizada en Twitter, donde por lo general existen una gran cantidad de errores en la escritura, abreviaciones etc.

Otro aspecto interesante, aunque no ha sido posible profundizar en gran medida es la velocidad con la que el traductor catalán-español alcanzaba buenos resultados en fase de entrenamiento. Tal y como señalan los autores en el artículo "Machine Translation of Very Close Languages" construir un traductor entre dos pares de lenguas cercanas como puede ser catalán-español es mucho más sencillo computacionalmente hablando que construir un traductor del español-ingles. En cierto modo, esto se puede deber a la cercanía sintáctica y semántica entre el par de lenguas a traducir.

Por último comentar que este trabajo a permitido profundizar en los problemas que surgen al tratar con redes neuronales. Principalmente aquellos problemas relacionados con la memoria de la GPU. Lo que obliga a explorar el tamaño de batch para dar con aquel que es adecuado para poder ser ejecutada sin tener problemas de memoria.

11 Posibles mejoras

Uno de los aspectos interesantes a la hora de construir traductores robustos es la necesidad de disponer de grandes volúmenes de datos que permitan estimar adecuadamente los parámetros del modelo. Debido a las limitaciones computacionales de la maquina donde he ejecutado los experimentos se ha tenido que utilizar un subconjunto reducido de los corpus. Una posible mejora pasaría por emplear mayores corpus de entrenamiento lo cual permitiría mejorar las prestaciones.

Siguiendo con el corpus de entrenamiento, se podría realizar una selección "inteligente" del subconjunto de entrenamiento que se va a utilizar para entrenar los modelos de traducción ya que no todos los pares de entrenamiento tienen la misma relevancia a la hora de traducir un tweet.

Por último, una de las partes más tediosas pasa por revisar las muestras traducidas por el sistema, para post-editarlas y utilizarlas en el ajuste de la red. En este proceso se podría implementar tal y como se ha comentado en las clases de teoría una selección inteligente de las muestras que va a revisar el operador, por ejemplo implementando una estrategia donde el operador solo tenga que revisar aquellas muestras que sean más inciertas para el sistema.