**Traducción automática**

Práctica 1 - Traducción estadística basada en frases: MOSES

# Objetivos

En esta práctica se pretenden construir sistemas de traducción utilizando pares de frases bilingües. Para este fin, utilizaremos la herramienta *Moses*. En la próxima sesión, se construirá un sistema similar basado en redes neuronales usando la herramienta NMT-Keras.

# Inicialización del entorno

En primer lugar, deberemos establecer una serie de variables de entorno que nos faciliten el uso y acceso a *Moses* y demás herramientas.

Graphical user interface, text

Description automatically generated

# Prueba de funcionamiento

Para comprobar que el entorno está correctamente configurado, se pretende traducir una frase con uno de los modelos pre-entrenados.

~/TA> mkdir Practica1; cd Practica1

~/TA/Practica1> mkdir Prueba; cd Prueba

~/TA/Practica1/Prueba> wget http://www.statmt.org/moses/download/sample-models.tgz

~/TA/Practica1/Prueba> tar xzvf sample-models.tgz

~/TA/Practica1/Prueba> cd sample-models/

~/TA/Practica1/Prueba/sample-models> moses -f phrase-model/moses.ini \

< phrase-model/in > out

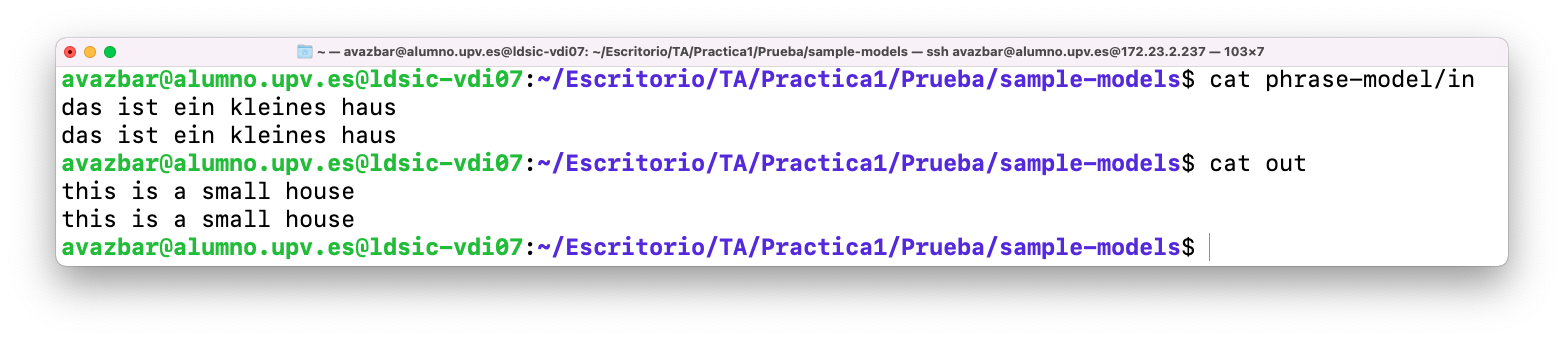
~/TA/Practica1/Prueba/sample-models> cat out

Graphical user interface, text

Description automatically generated

# Preparación de los datos

A continuación, utilizaremos un conjunto de datos de la tarea *EuTrans* de traducción castellano-inglés en el contexto de un turista frente al mostrador de un hotel.



El corpus ya se encuentra tokenizado, por lo que no es necesario someterlo a este proceso. No obstante, sí es necesario limpiar los datos para que las frases en sendos ficheros se alineen uno a uno. También será necesario crear un modelo de n-gramas para el idioma de destino, en este caso se ha apostado por un modelo de trigramas.

Graphical user interface, text, application

Description automatically generated

Graphical user interface, text, application

Description automatically generated

# Entrenamiento de los modelos de lenguaje de salida (inglés)

Generamos un modelo de trigramas como modelo de lenguaje. Más adelante será necesario disponer de la ruta absoluta al modelo generado, por lo que la exportamos como variable de entorno.

Text

Description automatically generated

Text

Description automatically generated

# Entrenamiento del modelo de traducción

Para construir la tabla de segmentos y modelos de reordenamiento, ejecutamos el siguiente comando que encontramos en el enunciado de la práctica:

>> $SCRIPTS\_ROOTDIR/training/train-model.perl -root-dir work \

-mgiza -mgiza-cpus $CPU \

-corpus training.clean -f es -e en \

-alignment grow-diag-final-and -reordering msd-bidirectional-fe \

-lm 0:3:$LM -external-bin-dir $GIZA>training.out

Graphical user interface, application, table

Description automatically generated

# Entrenamiento de los pesos del modelo log-lineal

Una vez obtenido el modelo de desarrollo, obtenemos los pesos del modelo log-lineal mediante MERT. Se ha limitado a 5 el numero de iteraciones para poder realizar la práctica en un tiempo razonable al mismo tiempo que evitamos una excesiva sobrecarga sobre las máquinas virtuales del departamento.

>> $MOSES/scripts/training/mert-moses.pl \

train/development.clean.es train/development.clean.en \

$MOSES/bin/moses train/work/model/moses.ini \

--maximum-iterations=5 \

--mertdir $MOSES/bin/

Text

Description automatically generated

# Proceso de traducción

Podemos comprobar mediante un conjunto de prueba, la calidad de las traducciones. Como puede observarse, la traducción, aun siendo gramaticalmente correcta, no respeta las normas de estilo del inglés, donde por ejemplo es común añadir “please” al final de la frase. Es comprensible pues, estamos haciendo uso de modelos probabilísticos para traducir.

# Graphical user interface, text, application Description automatically generated

# Evaluación del proceso de traducción

A pesar de las limitaciones, encontramos que el modelo tiene un BLEU bastante alto (algo más del 90%) por lo que aun no siendo similar a la traducción que haría un humano, es aceptable.

# Graphical user interface, text Description automatically generated

# Ejercicio 1

Probar el modelo obtenido en el apartado 6 sin ajuste de pesos.

El modelo se encuentra en: $MOSES/bin/moses -f / ../train/work/model/moses.ini < test.es > test.hyp

*Graphical user interface, text

Description automatically generated*

Al obviar el ajuste de pesos del séptimo apartado, se aprecia una ejecución más rápida al mismo tiempo que una caída sustancial del BLEU obtenido. Esto se debe a que el uso de MERT ayuda al ajuste de los pesos dado un modelo de desarrollo, aunque esto lo hace dependiente de este y añade coste computacional extra.

# Ejercicio 2

Probar para valores más altos del número máximo de iteraciones del MERT

*5 iteraciones (original)*

Graphical user interface, text, application

Description automatically generated

Graphical user interface, text

Description automatically generated

*10 iteraciones*

*Graphical user interface, text

Description automatically generated*

*50 iteraciones*

*Graphical user interface, text

Description automatically generated*

*100 iteraciones*

Graphical user interface, text

Description automatically generated

Con el doble de iteraciones la mejoría es muy leve y con cincuenta o cien apreciamos el efecto del sobre ajuste, perdiendo incluso calidad en la traducción.

# Ejercicio 3

Probar distintos valores de n-gramas (apartado 5)

Text

Description automatically generated

*Unigrama*

*Graphical user interface, application

Description automatically generated with medium confidence*

Graphical user interface, text, application

Description automatically generated

Con unigramas es imposible conseguir un modelo útil, ya que es como no tener modelo de lenguaje.

*Bigrama*

*Graphical user interface

Description automatically generated with medium confidence*

Graphical user interface, text

Description automatically generated

Con bigramas empeora el BLEU obtenido, esto se debe a la falta de historia pasada y contexto previo de las oraciones.

*Trigramas (original)*

Graphical user interface, text

Description automatically generated

*Cuatrigramas*

Graphical user interface

Description automatically generated with medium confidence

Graphical user interface, text

Description automatically generated

Con cuatrigramas también se experimenta una caída en el BLEU, esto es debido a que, a mayor numero de n-gramas obtenemos un menor numero de combinaciones no vistas en el entrenamiento.

# Ejercicio 4

Opcional: Probar MIRA para el entrenamiento de los pesos del modelo log-lineal (ver manual de Moses)

Según el manual de MOSES, cuando se proporciona el flag --batch-mira, se reemplaza la llamada a MERT por una a kbmira.

De esta forma, se activa el parámetro de tunnning k-best batch MIRA, que lleva a cabo online-training utilizando listas k-mejores agregadas como una aproximación del espacio de búsqueda.

<https://moses-support.mit.narkive.com/ZZ0UD01C/k-best-batch-mira-now-in-moses>

$MOSES/scripts/training/mert-moses.pl \

train/development.clean.es train/development.clean.en \

$MOSES/bin/moses train/work/model/moses.ini \

--batch-mira \

--maximum-iterations=5 \

--mertdir $MOSES/bin/

Graphical user interface, text, application

Description automatically generated

# Ejercicio 5

*Good Turing*

Para aplicar Good Turing basta con no especificar un método de suavizado en la llamada al script, al ser este el aplicado por defecto.

*Graphical user interface, text, application

Description automatically generated*

Graphical user interface, text

Description automatically generated

*Unmodified Kneser-Ney*

Graphical user interface

Description automatically generated with low confidence

Graphical user interface, text, application

Description automatically generated

El método de Kneser-Ney utilizado originalmente, ofrece mejores resulados que Good Turing y Unmodified Kneser-Ney. Esto es debido a que el primero utiliza un contador de los elementos vistos una vez para imputar una probabilidad a los elementos no vistos. Por otro lado, Kneser-Ney se basa en el contexto para imputar dicha probabilidad. En este sentido se comprende que la versión más refinada del método sea la que mejor funcione.

# Ejercicio 6

Opcional: Probar moses monótono (ver manual de Moses, página 67)

Graphical user interface, text

Description automatically generated

Graphical user interface, text, application

Description automatically generated

Para aplicar moses monótono, es necesario establecer el parámetro **-distortion-limit**

a cero, o lo que es lio mismo, no reordenar. Con esto se consigue una traducción literal. Es evidente que este tipo de traducción puede ser más rápida, pero penaliza el BLEU obtenido.

**Traducción automática**

Práctica 2- Traducción basada en redes neuronales recurrentes

**Ejercicio 1**

|  |  |
| --- | --- |
| **EMBEDDING\_SIZE** | **BLEU** |
| 32 | 87.74 |
| 64 | 94.00 |
| 128 | 97.70 |
| 256 | 98.20 |
| 512 | 97.10 |

**Ejercicio 2 (****EMBEDDING\_SIZE = 256)**

|  |  |
| --- | --- |
| **ENCODER/** **DECODER\_HIDDEN\_SIZE** | **BLEU** |
| 32 | 97.24 |
| 64 | 98.47 |
| 128 | 98.29 |
| 256 | 97.64 |
| 512 | 89.69 |

**Ejercicio 3 (EMBEDDING\_SIZE = 256, ENCODER/ DECODER\_HIDDEN\_SIZE = 64)**

|  |  |  |
| --- | --- | --- |
| **Adam** | **Adagrad** | **Adadelta** |
| 98.47 | 13.62 | 1 |

**Ejercicio 4**

|  |  |
| --- | --- |
| **MODEL\_SIZE** | **BLEU** |
| 32 | 63.51 |
| 64 | 87.50 |
| 128 | 90.25 |
| 256 | 86.63 |