Práctico 4 – Traducción estadística por frases

En este práctico usarán las siguientes herramientas que son el estado del arte en TA estadística para construir un sistema y de ser posible completar las tareas propuestas en la sección 1:

- GIZA++ para entrenar el modelo de traducción → ya lo tienen instalado
- SRILM para entrenar el modelo de lenguaje
- Moses decoder

1. Tareas

La idea es crear un sistema base y uno mejorado utilizando los mismos textos del práctico anterior y comparar los resultados, por lo tanto deberán hacer lo siguiente:

1. Crear un sistema baseline con el corpus de entrenamiento utilizado en el practico 3 (sys-1) y traducir el archivo de test con este sistema (ver sección 2)
2. Optimizar los parámetros de sys-1 con MERT y traducir con esta nueva versión (sys-2), ver mas detalles en la sección 2
3. Agregar más texto al corpus de entrenamiento y repetir 1. y 2.; los nuevos sistemas serán sys-1b y sys-2b.
    a. Los textos a agregar se pueden seleccionar del corpus Europarl http://www.statmt.org/europarl/ o del archivo http://www.famaf.unc.edu.ar/~pestrella/practico-3/dev.zip
4. Comparar la salida de los cuatro sistemas e indicar si alguna es mejor que otra, en ese caso cual y por que, es decir, hacer una evaluación (mas o menos informal) de los sistemas. Más adelante se evaluaran con métricas humanas y automáticas usadas en el área.
5. Mandarme por mail los archivos generados (las traducciones) y un informe del practico (indicando los problemas/soluciones encontrados durante el proceso, datos usados, resultado de la evaluación, etc)

NOTA: Para este práctico vamos a ir al laboratorio este viernes (21/05/2010) y para quien no pueda, no tenga tiempo o no tenga ganas de venir, en la sección 3 (mas abajo) le dejo algunas instrucciones para instalar las herramientas. Cualquier problema, duda, sugerencia, etc me mandan mail.

2. Creación de sistema baseline y optimización

Una vez instaladas estas cosas, hay que crear el sistema baseline (conviene crear directorios separados donde dejar el corpus, crear el modelo y optimizar)

1. Pre-procesamiento de texto
    a. `mkdir corpus`
    b. Tokenizar el texto
    ```
    scripts/tokenizer.perl -l fr < corpus.fr > corpus.tok.fr
    scripts/tokenizer.perl -l en < corpus.en > corpus.tok.en
    ```
    c. Lowercase training data
    ```
    scripts/lowercase.perl < corpus.tok.fr > corpus.lc.fr
    scripts/lowercase.perl < corpus.tok.en > corpus.lc.en
    ```
2. Crear modelo de lenguaje (en nuevo directorio ml/)
    a. Usar SRILM
    `/path/bin/i686/ngram-count -order 5 -interpolate -kndiscount -text corpus/europarl.lc -lm ml/modelo.lm` → nombre y donde crearlo
3. Entrenar el modelo de traducción
    a. `bin/moses-scripts/scripts-YYYYMMDD-HHMM/training/train-factored-phrase-model.perl -scripts-root-dir bin/moses-scripts/scripts-YYYYMMDD-HHMM -root-dir working-dir -corpus corpus/corpus.lc -f fr -e en`

```
-alignment grow-diag-final -reordering msd-bidirectional-fe
-lm 0:5:working-dir/lm/modelo.lm:0
```

   b. Esto tarda bastante de acuerdo a la cantidad de texto que se use; eventualmente podrían `nohup <comando>`

4. Optimización de parámetros
   a. Para este proceso dejar una porción del
   b. Tokenizar el texto que se usara en la optimización
   ```
   mkdir -p working-dir/tuning
   scripts/tokenizer.perl -l fr < wmt08/dev/dev2006.fr > working-
   dir/tuning/input.tok
   scripts/tokenizer.perl -l en < wmt08/dev/dev2006.en > working-
   dir/tuning/reference.tok
   ```
   c. Poner en minúsculass
   ```
   scripts/lowercase.perl < working-dir/tuning/input.tok > working-
   dir/tuning/input
   scripts/lowercase.perl < working-dir/tuning/reference.tok > working-
   dir/tuning/referente
   ```
   d. Ejecutar el script para mert con nohup porque tarda mucho en ejecutarse (incluso días dependiendo de la cantidad de texto)
   Note that this step can take many hours, even days, to run.
   ```
   nohup bin/moses-scripts/scripts-YYYYMMDD-HHMM/training/mert-moses.pl
   working-dir/tuning/input working-dir/tuning/reference moses/moses-
   cmd/src/moses working-dir/model/moses.ini --working-dir working-dir/tuning
   --rootdir bin/moses-scripts/scripts-YYYYMMDD-HHMM
   ```
   e. Cuando termine el proceso, agregar los nuevos lambdas en en archivo `moses.ini`
   ```
   scripts/reuse-weights.perl working-dir/tuning/moses.ini < working-
   dir/model/moses.ini > working-dir/tuning/moses.weight-reused.ini
   ```
   f. Traducir usando `train-factored-phrase-model.perl` con el nuevo archivo de configuración de moses

3. Instalación de herramientas

1. Instalar SRI-LM (necesita gcc version 3.4.3 o mas nueva)
   a. Bajarlo de http://www.famaf.unc.edu.ar/~pestrella/srilm.tar o de http://www.speech.sri.com/projects/srilm/download.html
   b. Cambiar la variable del Makefile para que apunte al directorio correcto
   Ej. SRILM = /home/estrella/srilm-1
   c. Algunas otras variables que probablemente tengan que cambiar ncluyen: MACHINE_TYPE, CC, CXX, TCL_INCLUDE y TCL_LIBRARY deben tener los paths correspondientes o dejarse vacias y setear NO_TCL=X
   d. Les recomendaría que instalen las cosas que siguen según les haga falta
      i. Para cada plataforma especifica deben consultar los archivos en doc/README.<os>-<machinetype>, (ej doc/README.windows-cygwin)
      ii. Según la documentación de SRILM, para compilar hacen falta los siguientes paquetes: gcc versión 3.4.3 o mas, GNU make, Tcl toolkit versión 7.3 o mas nueva, awk (gawk), gzip, bzip2, p7zip
   e. En el directorio raíz hacer
   `make World`
   y para compilar en maquinas mas “raras” poner el tipo de maquina en el comando
   `make MACHINE_TYPE=i686-m64 World`

f. `make test` y `make cleanest` corren una serie de pruebas a ver si todo esta bien instalado y borran cosas que no se usan, respectivamente

2. Instalar Moses
   a. Bajarlo del repositorio svn (hacer check out)
      mkdir -p moses
      svn co https://mosesdecoder.svn.sourceforge.net/svnroot/mosesdecoder/trunk moses
   b. Compilar con
      cd moses
      ./regenerate-makefiles.sh
      ./configure --with-srilm=/path-to-srilm
      make
   c. Instalar scripts adicionales que se usan para ajustar parametros (tuning) y decodificar. Editar el Makefile para setear las variables targetdir y bindir a los directorios correspondientes

```
mkdir -p bin/moses-scripts

###Editar moses/scripts/Makefile
TARGETDIR=/path-absoluto/bin/moses-scripts
BINDIR=/path-absoluto/bin
###

cd moses/scripts/
make release
```

      El resultado de esta operación debería mostrarles este mensaje

```
export SCRIPTS_ROOTDIR=/full-path-to-workspace/bin/moses-scripts/scripts-YYYYMMDD-HHMM
```

   d. También se pueden bajar estos scripts adicionales para tokenizar y generar archivos xml con el formato adecuado para la evaluación que haremos mas adelante
      http://www.statmt.org/wmt09/scripts.tgz
   e. El binario del decodificador debería generarse en /moses-cmd/src/

En estas páginas hay mucha mas información sobre la utilización de estas herramientas (pero en inglés)

1. Por donde empezar http://www.statmt.org/moses/?n=Development.GetStarted
2. Un tutorial http://www.statmt.org/moses/?n=Moses.Tutorial
3. Creación de un sistema base http://www.statmt.org/wmt09/baseline.html (base de este documento)