Universidad Nacional de Rosario

Deep Learning

Trabajo Práctico IV

Integrantes:
Crosetti Mariano
Tenaglia Felipe
Villagra Martín
Zanetti Álvaro

Introducción

En la primer parte utilizamos el dataset wikiCorpus en español para entrenar una red LSTM que genere texto en castellano, prediciendo el siguiente caracter más probable a partir del texto ya generado o de una semilla (en las primeras iteraciones).
El wikiCorpus se puede encontrar en el siguiente enlace: `http://www.cs.upc.edu/~nlp/wikicorpus/`.
El código base para la creación del dataset, el cual luego fue modificado para realizar distintos tipos de preproceso, se puede encontrar en: spanish.py.
El código para la creación de la red LSTM se puede encontrar en: lstm_text_generation.py.

En la segunda parte combinamos la red convolucional del trabajo práctico 3 (para reconocer caracteres manuscritos) con la LSTM, de manera de reconocer líneas manuscritas completas. La evaluación se realizará sobre imágenes de oraciones manuscritas del texto de Borges, "El Aleph".

Primera parte

Explicación de los módulos

A continuación una breve guía de cada módulo:

- `spanish_con_dict_7500lines.py`: Aplica el preprocesamiento al dataset y lo guarda en un *.h5*.
- `train_lstm.py`: Sirve para entrenar la red LSTM y guardar los pesos.
- `lstm.py`: Provee una interfaz para utilizar una LSTM ya entrenada (carga modelo y predice).
- `utils.py`: Define funciones utiles como el reemplazo de los acentos por sus versiones sin acentuar.

Ejercicio 1.a

A partir de 11 iteraciones, el texto en español generado por la LSTM (base) empieza a tener sentido. Sin embargo, esto está sujeto al diversity utilizado, como se explicará en el apartado *d*.

```
Iteration 11
Epoch 1/1
470462/470462 [==============================] - 209s - loss: 1.5567

----- diversity: 0.2
----- Generating with seed: "andes peñazcos al mar. en tiempos del im"
andes peñazcos al mar. en tiempos del importante de la provincia de la
partida de la comunidad de la provincia de la provincia de la comunidad de
la periodisión de la provincia de estados unidos de la considera de la
provincia de la provincia de la provincia de la considera de la provincia
de la comuna de la canción de la provincia de la provincia de la provincia
de la provincia de la pinterrata de la provincia de la comunidad de la pro

----- diversity: 0.5
----- Generating with seed: "andes peñazcos al mar. en tiempos del im"
andes peñazcos al mar. en tiempos del importante munichinde con los árbiti
y antes de estados unidos en cuando después de la provincia de un plomptor
por fueron al pueblo artista la comienza al estado al concelto de las
lenguas entre el compaño comandante en el contra un resumperio de desde en
el manti, y en un año 2000 sobre el ley de la carrera de la canción de la
estación de la interpreta de los partidos a la columna de la provincia

----- diversity: 1.0
----- Generating with seed: "andes peñazcos al mar. en tiempos del im"
andes peñazcos al mar. en tiempos del imadero 9  ese le cuenta encuentro
aerzen, espiracional de perrodor después de el medio de colegio, riflecfe
es no le continuó lleman poco estrelando como camino y que entreva de 3
límetres en intespkárabera y y del rey quotherhatón salzape la aña. feneró
buegó la 24 de la piorez funcional: contimión seche la tiromóda de
estaformálidebre chaiolan y eña provincia por podrín su bergena ayorroel atritc

----- diversity: 1.2
----- Generating with seed: "andes peñazcos al mar. en tiempos del im"
andes peñazcos al mar. en tiempos del imanba,fue que fran para rusar-dar cería
aterbitab 4 por tradecer, y complez meyoronde el instrumento oriva corr "ler :
asemporó macaraga emilizar conceptado. tienen: puo dubslives ero fárcular que
el coso simplesán, pantiovina dos dil) el agus jhtu 1969). genentaba guajacion
por la heberti wesco no incluidm se ciudael, bopggsosh, ctm, estos fotes y que
mamenta del helgio no hom del cormono linos a
```

Ejercicio 1.b

Podemos pensar el loop existente dentro de una LSTM, el cual permite pasar información de un paso de la red al siguiente, como la misma red desplegada n veces (donde los input son datos secuenciales de dimensión n), como se observa en la figura 1.
Por lo tanto, la tamaño de la red se multiplica n veces, haciendo que esta sea mucho más

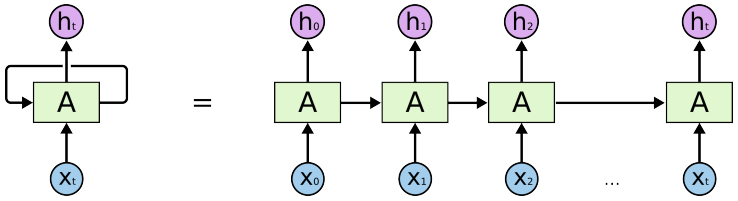

Figura 1: Una RNN desplegada

costosa de entrenar.

Se intentó ajustar la cantidad de inputs (es decir la cantidad de caracteres anteriores que tiene en cuenta la LSTM para predecir el siguiente caracter). Este meta-parámetro cambia radicalmente el tamaño de la red (como es esperable), pero reducirlo produce que la predicción contenga más bucles y repeticiones, aún con diversidades altas.

Ejercicio 1.c

Como primera medida, al conjunto de datos utilizado se le eliminaron los acentos. Decidimos hacer esto ya que la CNN del trabajo práctico anterior no fue entrenada con estos caracteres, por lo que nunca serán predichos. Los caracteres fueron reemplazados por sus correspondientes sin acentuar.

Para el preprocesamiento de los datos (wikiCorpus) se probaron los siguientes métodos:

- Filtrar todas las líneas del corpus que contengan caracteres especiales, como: ^{a}, ^{2}, ^{3}, °, β, ð, Ø, entre otros.
- Filtrar todas las líneas que contengan palabras que no se encuentren en los diccionarios: diccionario-espanol.txt y diccionario-espanol2.txt.
- Filtrar las líneas que contengan alguna de las palabras menos frecuentes (30 %).

Al final utilizamos los primeros dos métodos, ya que a simple vista pudimos observar una mejora en el texto generado.

El código para generar el dataset final que se usó para entrenar la LSTM se puede encontrar en el módulo `spanish_con_dict_7500lines.py` el cual tomará los datos desde

`./src/spanish-rawdata` (aquí deben estar los archivos de wikiCorpus en español) y generará el archivo `./src/wikicorpus/con_dict_7500lines.h5` que contiene el dataset.

Ejercicio 1.d

La variación de temperatura (o diversity) permite modificar la rigidez del modelo. Se puede observar que a muy baja temperatura la red genera texto con bucles, repitiendo frases una y otra vez, sobre todo construcciones como "de la *{sustantivo}* de la *{sustantivo}* de la ...". Por otro lado, a muy alta temperatura la red adquiere demasiada flexibilidad y comienza a predecir palabras muy alejadas de cualquiera conocida y frases sin estrucutra aparente. Los mejores resultados los obtuvimos con valores intermedios, entre 0,7 y 0,9.

También probamos aumentar la temperatura justo en la predicción de la primer letra de las palabras, pero este parámetro adicional era difícil de ajustar, pues algunas veces se tenían buenos resultados para una semilla pero malos para otra.

Red LSTM final

Resultados con la LSTM final aplicando los preprocesos mencionados:

```
----- diversity: 0.6
----- Generating with seed: "erta original, ha sido substituida recientemente
por una de lonas tensadas para proporcionar una mej"
erta original, ha sido substituida recientemente por una de lonas tensadas
para proporcionar una mejor ciudad en el control de traslado de la compania
de la comuna de la segunda guerra mundial. el portero de bayara fue superada
por el centro de la seleccion de futbol de plata del control de la ciudad,
de transporte con la historia al transporte de san martin en la iglesia de
la tierra y con una tierra antigua, aunque el presidente fue construido en
bolivia. la corona se ha construido en algunos

----- diversity: 0.7
----- Generating with seed: "erta original, ha sido substituida recientemente
por una de lonas tensadas para proporcionar una mej"
erta original, ha sido substituida recientemente por una de lonas tensadas
para proporcionar una mejor actuacion de la energia de la escena en la
pelicula a castilla en 1916 en algunos restaurantes y la materia no estan
formadas, habiendo proclamado algunas alturas de la compania de la construccion
de un agua que produce este estado de competicion y de la que cumple la
primera resistencia escrita, de inicio el actual buen sudeste de la cuenca
```

```
puesta de ciudad para ser un periodo de ser analizado

----- diversity: 0.8
----- Generating with seed: "erta original, ha sido substituida recientemente
por una de lonas tensadas para proporcionar una mej"
erta original, ha sido substituida recientemente por una de lonas tensadas
para proporcionar una mejor primavera semana importante de vida de individuos,
invitando a un periodo de corriente en el sistema de hijas, o aereo, osferia
y al elegido por ferrocarril de chile. con la edicion del comportamiento de
buenos aires, se encuentra en la armada municipal de la serie de la escuela
mexicana, al gobernador de la marina de alemania, en la region de puerto
personaje, al horno de pagas dirigente de la

----- diversity: 0.9
----- Generating with seed: "erta original, ha sido substituida recientemente
por una de lonas tensadas para proporcionar una mej"
erta original, ha sido substituida recientemente por una de lonas tensadas
para proporcionar una mejor antiguedad de retroceso. un asistente fue un
batallan de nueva escolas. que es un control y en la universidad tales como
la descontrajora peruana. en sus aguas les impresa sus propios duasor. las
primeras licencias de constitucion es uno de los procedimientos de la compania
de las distancias de la y de europa. el objeto son arribo para disco
localizados en el campo. 1972 se produjo en buenos ai

----- diversity: 1.0
----- Generating with seed: "erta original, ha sido substituida recientemente
por una de lonas tensadas para proporcionar una mej"
erta original, ha sido substituida recientemente por una de lonas tensadas
para proporcionar una mejor fecha asi para ser defendidas sobre el interes
en la barts, despues de estos sea. inpenson carbas anarquista del piex metal
melviriso y seitirico y juita de cada lemas donde seguiria el pasuo de barras
casamitae para dice que estos tiempos encuentrarlos se dicas eficientes de
5 anos despues realmente cuyo especie se harian ningunos tipos de abusos.
a partir de 1964, no esta situada con particul

----- diversity: 1.1
----- Generating with seed: "erta original, ha sido substituida recientemente
por una de lonas tensadas para proporcionar una mej"
erta original, ha sido substituida recientemente por una de lonas tensadas
para proporcionar una mejor punta que deben laborar en el analisis battastid
```

```
y por eso evidente. es decasta de femeniencias largas acelo numerosas regiones
huarcurales, y remaban un tiempo altrojo seces para n6dy, y desde los cuales
tambien se african significa dober palaje. ocasion es que la noche se hectina
a traves de umbil para elementos mineridaes como sexual blancos, bostones
extratos economicos. siendo su ceistraci

----- diversity: 1.2
----- Generating with seed: "erta original, ha sido substituida recientemente
por una de lonas tensadas para proporcionar una mej"
erta original, ha sido substituida recientemente por una de lonas tensadas
para proporcionar una mejor. la enimificacion  soberonaria extraceras.
biocesaico solpa ensayencia, 1828. jose de japara publica una nacion. durante
la ildca la java irlandesa, 1984. surfon sin embargo por los e dolores,
oltiodsu. unos anos ante las elecciones puestos yor. la rubina es negros
remodelaciones de la fundacion. de carga fuera ksii gago decadio nuevo una
encuentro de caga cirumente por the paredan, suhiccuks y
```

La arquitectura final, que luego utilizamos en la segunda parte, es la siguiente:

```
model = Sequential()
model.add(LSTM(128, name='lstm1-128', consume_less='gpu', input_shape=(maxlen,
     len(chars)), return_sequences=True))
model.add(Dropout(0.2))
model.add(LSTM(128, name='lstm2-128', consume_less='gpu', return_sequences=True))
model.add(Dropout(0.2))
model.add(LSTM(256, name='lstm3-256', consume_less='gpu', return_sequences=True))
model.add(Dropout(0.2))
model.add(LSTM(256, name='lstm4-256', consume_less='gpu'))
model.add(Dropout(0.2))

model.add(Dense(256, name='densa_extra'))
model.add(Dropout(0.3))
model.add(Dense(len(chars), name='softmax', activation='softmax'))
```

La red entrenó durante 4 días, no de manera continua sino que se comenzó con un `lr=0.01` y se fue reduciendo hasta un `lr=0.0001`, con el mejor resultado en la iteración 49 de la última ejecución. Este código se puede encontrar en el módulo `train_lstm.py` que tomará el dataset para entrenar desde `./src/wikicorpus/con_dict_7500lines.h5`.

Segunda parte

En la presente sección se explicará como se hizo uso de la CNN + LSTM para la predicción del texto.

Nuestro primer intento fue sobre el dataset provisto, pero debido a que la caligrafía difería bastante de la que se usó para entrenar la CNN (la mayoría de aquel dataset estaba en imprenta) y la presencia del renglón, la cátedra ha sugerido crear nuestro propio conjunto. Este lo podemos encontrar en la carpeta `src/newdataset` y es el que se ha utilizado a lo largo del trabajo.

La idea es usar la LSTM para corregir lo que la CNN reconozca, con el conocimiento del español (como una sugerencia de la palabra de la que puede tratarse). Creemos que esto es mucho más eficaz que usar otros métodos (como heurísticas que usen diccionario) porque la LSTM usará información del contexto en el cual está incluida la palabra, además de manejar de manera flexible (pero potente), por ejemplo, a las variaciones de conjugaciones o formaciones complejas que se pueden presentar (como número y género de adjetivos o sustantivos).

Instrucciones de uso

Para obtener los resultados del trabajo:

- Ejecutar `gennpy.py` para generar las predicciones de la CNN.
- Ejecutar `prediccionesfinales.py` para generar la predicción final de cada oración.

El programa asume que el formato de la imagen es: "Linea XX - " ++ *texto_linea* ++ ".ext", donde XX son 2 dígitos decimales que indica el número de la línea, ".ext" la extensión de la imagen y *texto_linea* es el texto original que debe ser producido (si se dispone de él, es útil para luego medir y comparar que tan buenos son nuestros resultados).

Los resultados son almacenados en `/predictions`, donde se genera, para cada línea, un archivo cuyo nombre es el nombre original de la imagen. Además, en la carpeta `/preview` se encuentran algunas previsualizaciones y resultados parciales:

- El resultado del preprocesamiento de cada imagen (lo hace el módulo `preproceso.py`).
- Las ventanas para cada imagen en la subcarpeta `/win` (lo hace el módulo `genventana.py`).
- La predicción de caracter más probable para cada ventana procesada y mediciones respectivas (lo hace el módulo `cnn.py`).

Explicación de los módulos

A continuación una breve guía de cada módulo:

- `paths.py`: Define los paths globales en los que se basan los otros módulos (está por si cambia la estructura de directorios de nuestro proyecto).
- `preproceso.py`: Contiene los métodos que se ocupan de preprocesar –al igual que se hizo con los datos que se usaron para el entrenamiento de la CNN– las líneas del texto de Borges adecuadamente. Además se realizan tareas como el recorte de las líneas. El método que se utiliza desde el exterior es `preprocessOne(fileName)` que devuelve una imagen luego del preproceso. Las imágenes preprocesadas son guardadas en la carpeta `/preview`.
- `genventana.py`: Dada una imagen genera una lista de imágenes que son los recortes de desplazar una ventana a lo largo de la imagen. De ello se encarga `windows(nameImg,winWidth,step,finalWidth)`, que es la función que se utiliza desde el exterior. Las ventanas de cada imagen se guardan en subcarpetas en `preview/wins` (una carpeta por línea).
- `cnn.py`: Contiene la función `applyCNN(fileName)` que dado un path de una imagen produce una matriz de predicciones (para cada ventana, la probabilidad de cada letra).
- `gennpy.py`: Aplica la CNN a las imágenes y guarda las predicciones en archivos .npy en `/npyCNN`. En `preview/predicciones.txt` se guarda la predicción del caracter más probable para cada ventana procesada.
- `costprediction.py`: Provee tres funciones de costo que miden qué tan buena es la predicción de la CNN.
- `lstm.py`: Provee una interfaz para utilizar la LSTM (cargar el modelo y predecir).
- `combinacion.py`: Tiene una lista (combinadores) que contienen pares `(nombreCombinador, funcionCombinadora)` donde `funcionCombinadora` es una función que toma alguna matriz de predicciones de la CNN y, haciendo uso de la LSTM, produce una predicción.
- `finalpredictions.py`: Ejecuta los combinadores y produce las predicciones. También mide la distancia de edición de cada combinador.

Preprocesamiento y generación de ventanas

El preproceso y la generación de ventanas se realiza en los archivos `preproceso.py` y `genventana.py`, respectivamente.

Al haber creado nuestro propio dataset, no fue necesario redefinir los límites de la imagen y recortarla, ya que la entrada estaba correctamente ubicada. El preproceso consta de 3 operaciones simples: aumento de contraste, escalado y, en este caso, consideramos apropiado agregarle a derecha e izquierda un márgen blanco para que, al generar las ventanas, el primer caracter se encuentre centrado. En la figura 2 podemos observar el resultado del preprocesamiento. La función `preprocessOne(f)` de `preproceso.py` es la encargada de esto, toma un nombre de archivo y devuelve un par `(nombreArchivo, imagenProcesada)`.

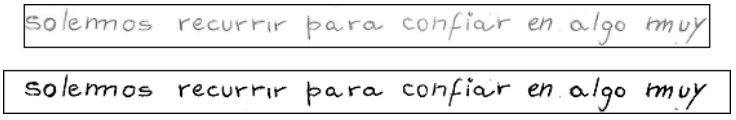

Figura 2: Arriba: Imagen original. Abajo: Imagen procesada. Podemos observar el aumento de contraste y el padding blanco a derecha e izquierda.

La generación de ventanas se realiza en `genventana.py`, función `windows(nameImg, winWidth, step, finalWidth)`. Toma un par `(nombre, imagen)` (el resultado del preproceso), el ancho de la ventana, el salto entre una ventana y la siguiente y el tamaño final de la imagen. `winWidth` podría ser menor a `finalWidth`, rellenándose el resto con blanco. La idea original era aislar el caracter con una ventana más angosta, pero los resultados no fueron satisfactorios. La función devuelve un par `(nombre, wins)`, donde `nombre` es el nombre del archivo argumento (arrastrado por motivos de debug) y `wins` es un arreglo con las ventanas generadas. En la figura 3 se puede observar un subconjunto de las ventanas generadas para la línea de la figura 2 (`step` = 2).

Cuantificando los resultados la CNN

Consideramos necesario contar con medidas de los resultados parciales, especialmente en proyectos como este, que constan de diferentes componentes ensamblados, de modo de poder cuantificar por separado la labor de cada uno y poder descomponer el análisis, en lugar de orientar todo al resultado final.

Es por esto que creamos tres diferentes medidas (en `costprediction.py`) para ver si el resultado de aplicar la CNN a la secuencia de ventanas de imágenes es bueno o no con más objetividad y precisión que una simple observación. Para la implementación de las

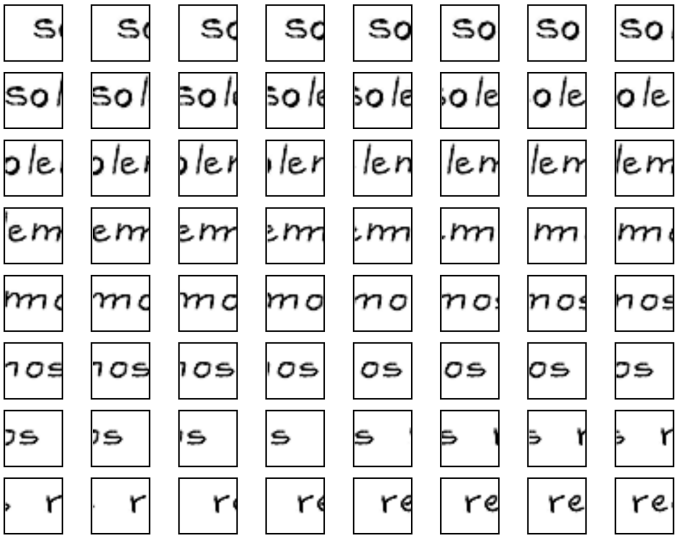

Figura 3: Ejemplo de las ventanas generadas.

medidas se utilizó programación dinámica. En el código se encuentra una justificación informal de con qué criterio se implementaron estas funciones.

Podemos ver que los textos que poseen un menor costo generan un mejor resultado final, por lo que creemos que las funciones elegidas son buenos indicadores de la performance de la CNN. A continuación comparamos los resultados de dos frases con costos altos y bajos (los combinadores serán explicados en la siguiente sección):

```
ORIGINAL: El Aleph_ - repetí.
PREDICCION CNN: EEEEEEEEEEEE lIIIIII       ’’’  PAAAAAAAAtttlllllllllleeeee
tfffffpoooot ttttthhhshh            ’’    irrrrrrreeeeeeee  fffffooop
eeeeeeee   ttttttFEEiiiii;iiiii..."

COSTO 1: 419.161676647
COSTO 2: 43.7125748503
COSTO 3: 532.452856289

combinador1: :E lI Atle fpo th re fo e ti.
EDIT-DISTANCE: 15
combinador2: E lI Atle fpo th re fo e ti.
EDIT-DISTANCE: 14
------------------------
```

```
ORIGINAL: sótano había un Aleph. Aclaró que un Aleph es
PREDICCION CNN: sssssssss ooooooooo   ttttttteEE  ooooaaaaaaaa
nnnnnnnhhooooooooooo        ’’’ttttthhhhhooooooaaauua
bbbbbbbbbbiiiiiiiiiaaaaaa..ne    vvvvvuuuuunnnnns,nn    t AAAAAAAAA
lllllllleeee fftfpfobbb  ttthhhhhhh ........
PPAAAAAAHHficccccclllllllllaaaaaaauarrrrrrroooooosooo
ggogggqqguuuuuvuvreeeeee  vvvvvuuuunnnnnnnn       pPAAAAAAAAHlMMllleeeee
tftpppbb ttthhs,h         ’’ eeeeeeeeeessssss

COSTO 1: 335.0
COSTO 2: 31.0
COSTO 3: 475.602025951

combinador1: so t oa no thoa bia vun Ale f th . Aclaro gue vun Alepth es
EDIT-DISTANCE: 16
combinador2: so t oa no thoa bia vun Ale f th . Aclaro gue vun Alepth es
EDIT-DISTANCE: 16
------------------------

ORIGINAL: solemos recurrir para confiar en algo muy
PREDICCION CNN: sssssssooooooolllllleeeeermmmrmmmmmmmooooooooaossssos
rrrrrrreeeeeccccccvvuuuuuurrrrrrrrrriiiirrrrrrr
ppppppooaaaaaaaaurrrrrrrraaaaaeen-  ccccccooooooornnnnn
fffffiiiiiaaaaiiirrrr-rr   eeeeeennnnnn...   aaaaaaaallllggggooooooo
rrrrrmmmmmmnnuuuuuuuyyyyy

COSTO 1: 113.20754717
COSTO 2: 5.66037735849
COSTO 3: 291.569259127

combinador1: solemos recurrir para confiair en. algo muy
EDIT-DISTANCE: 2
combinador2: solemos recurren r para confianzar r en. algo muy
EDIT-DISTANCE: 9
------------------------

ORIGINAL: Sí, el lugar donde están, sin confundirse, todos los lugares
PREDICCION CNN: ’sssSSSsiiii,,,,   eeeeeeellllll
T’lllliiiiggggoooaaaaaarrrrsr   ddcdddooooooonnnnnnnddddddldeeee
eeeeeeeessssssstttttttttaaaaaaaaannnnnnhn,,;,      ’’sssssiiiiiinnnns.M
```

```
ccccccccooooooonnnnnnnnnfffffffouuuuunnnnnn
ddddddiiiiiiirrrrrsssssssseeeeeee,,,   ttttttFooooodddddooooooosssss
lllllloooosssss lIlvlluuuuuggggaaaaaartteeeersss

COSTO 1: 138.235294118
COSTO 2: 32.0588235294
COSTO 3: 294.696532868

combinador1: Si, el ligoar donde estan 'sin confundirse, todos los lugares
EDIT-DISTANCE: 4
combinador2: Si, el ligo ar donde e estan 'sin confundir dirsen, todos los
lugares es
EDIT-DISTANCE: 15
```

Combinación LSTM + CNN

Fue un desafío y a la vez una oportunidad para probar diferentes alternativas. En primer lugar lo que observamos fue que el texto producido por la CNN se asemejaba a lo que se quería predecir (haciendo un análisis meramente visual) excepto por las letras repetidas y alguna que otra letra "colada". Lo primero es inmediato del uso de la ventana deslizable: un caracter es capturado muchas veces por la ventana. Lo segundo ocurre cuando la ventana está en la mitad de dos caracteres (por lo cuál es una ventana inútil) y la predicción es sobre caracteres "cortados".

Para evitar caracteres repetidos consideramos escribir a la salida sólo cuando la CNN registra un caracter como muy probable y no escribiendo repetidos a menos que la LSTM lo sugiera (de este modo podemos escribir caracteres que es natural que estén repetidos en el español como dobles "r" o dobles "l").

A la hora de elegir el caracter a escribir fue difícil encontrar una técnica que tenga en cuenta no sólo una ventana en particular sino un conjunto de ventanas. Es por eso que se eligió guardar un arreglo de probabilidades acumuladas y cuando éstas superaban cierto umbral, escribir dicho caracter y resetear el arreglo. De este modo queremos, mediante un modelo simple y rápido, capturar el hecho de que el caracter escogido deriva del análisis de muchas ventanas.

El `combinador2` trata de darle más relevancia a la LSTM, aunque tiene resultados notablemente inferiores, es curioso que en algunas ocasiones genera predicciones que sólo con la CNN no hubieran sido posible, como es el caso de agregarle una "z" a "niñez":

```
ORIGINAL: en la niñez, antes de la edad escolar. La escalera del
combinador1: en la nine, oarites de la edad escolar. Loa escaleran del
EDIT-DISTANCE: 6
combinador2: en la ninez , oarites de la edad escolar. Loa escalaera n del
EDIT-DISTANCE: 8
```

Restaría analizar como se podrían tener los beneficios de la LSTM y no afectar los resultados.

Finalmente probamos `simpleCombinador`, que es un generador de combinadores parametrizado, que sintetiza las técnicas que creemos que funcionaron de nuestras primeras pruebas y podemos ver los resultados para diferentes elecciones de parámetros, aunque `combinador1` sigue siendo, por ventaja, muy superior a las combinaciones de parámetros probadas.

Restaría, también, probar otras formas de incluir la LSTM, por ejemplo, sobre la predicción final, se podría tratar de modificarla de algún modo teniendo en cuenta la LSTM (esto no lo hemos probado).

Conclusiones

Si bien hemos rearmado el dataset para que se parezca más al conjunto de datos con los que se entrenó, los resultados que hemos obtenido han sido alentadores, aún cuando la CNN haya sido entrenada con un conjunto de datos de muchas clases, y con varias de esas clases con pocos ejemplos, dónde difícilmente las muestras que usamos para entrenar tengan la misma distribución que los caracteres de los textos que luego leemos. Se ha obtenido un costo total (medido en edit distance / caracteres totales) mínimo de: 16.78 % para el `combinador1`, que es el que mejor funciona.

Queda como trabajo pendiente encontrar combinadores mejores y con más fundamento o técnicas para el uso de la LSTM que contribuyan en mayor medida a ayudar a las falencias de la CNN.