Predicción de puntuación y capitalización en español con RNN bidireccionales

DEPARTAMENTO DE COMPUTACION
Facultad de Ciencias Exactas y Naturales - UBA

Leonel Braginski
LU: 385/21

Tobías Moraut
LU: 1507/21

Bianca Bramati
LU:1893/21

Damian Care
LU: 875/02

León Herrera
LU: 105/18

mails: leobraginski@gmail.com
tobiasmoraut7@gmail.com
damianos.care@gmail.com
biancabramati2@gmail.com
leonazulado@gmail.com

Introducción

La predicción automática de puntuación y capitalización es fundamental en tareas como la transcripción de voz, donde los modelos suelen generar texto plano sin signos de puntuación ni uso correcto de mayúsculas. Este procesamiento es esencial para mejorar la legibilidad y comprensión del texto en aplicaciones como subtitulado automático, asistentes virtuales y sistemas de dictado.

En este trabajo abordamos el problema utilizando tres métodos: uno clásico y dos basados en redes neuronales recurrentes (una RNN simple y una RNN bidireccional), y comparamos su desempeño en ambas tareas.

Tarea

Dado un string de texto en minúscula (de tamaño no determinado) y sin puntuación, se desean predicciones para cada token de puntuación y capitalización:

- **Puntuación:**
  ¿ : signo de pregunta de apertura
  , : coma
  . : punto (de división de oraciones)
  ? : signo de pregunta de cierre
  0 : sin puntuación

- **Capitalización:**
  0: todo en minúsculas (“hola”)
  1: primera letra en mayúscula (“Hola”)
  2: algunas letras en mayúscula (“McDonald's” o “iPhone”)
  3: todo en mayúsculas (ej.:“ONU”,“NASA”,“UBA”)

Ejemplo de instancia

	el	maestro	dice	el	inspector	es	un	ig	##nor	##ante
Punt. Ini.	0	0	0	0	0	0	0	0	0	0
Punt. Fin.	0	,	0	0	,	0	0	0	0	.
Cap.	1	0	0	0	0	0	0	0	0	0

Creación del Dataset

Se recopilaron datos de diversas fuentes, incluyendo Wikipedia, novelas y otros recursos textuales en español.

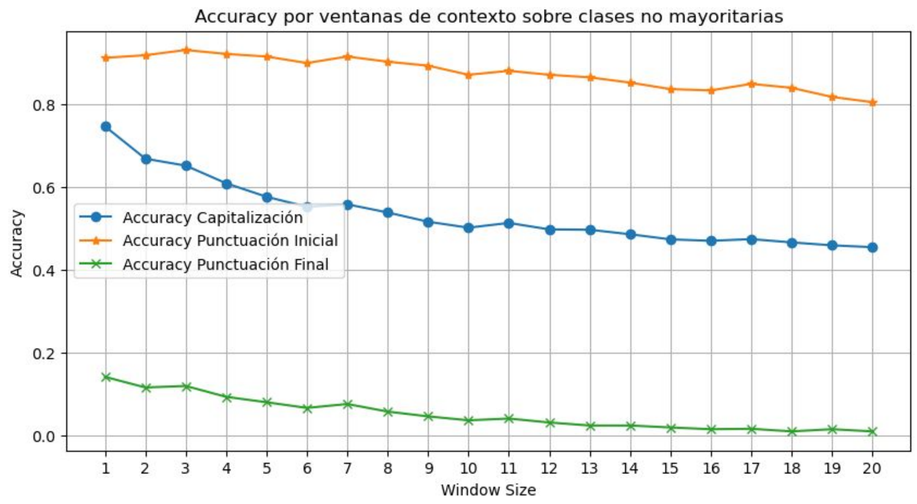
El proceso incluyó una fase de normalización del texto, donde se eliminaron caracteres no deseados y se estandarizó el formato. Posteriormente, se generaron muestras de texto a nivel de oración, desprovistas de mayúsculas y signos de puntuación. Cada muestra fue cuidadosamente anotada a nivel de token, proveyendo un conjunto de datos robusto y limpio.

- **Novelas: #5390 or.**
- **Wikipedia: #5739 or.**
- **Preguntas: #1500 or.**
- **Total: #12629 or.**

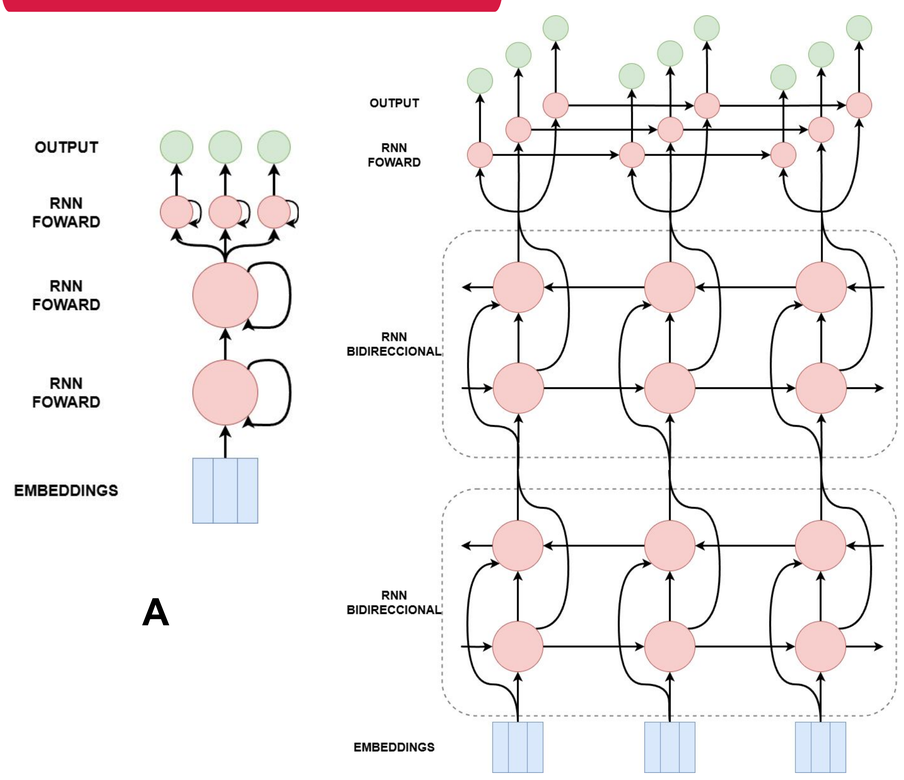
Modelo Clásico

Nuestro problema requiere predicción en secuencias donde nos importa el contexto en la secuencia. Los modelos clásicos no logran captar esto. Experimentamos con una implementación de **Random Forest** para cada problema de predicción. Agregamos artificialmente contexto usando de features ventanas de contexto, por cada instancia los features son sus *W* tokens anteriores y consecuentes. El siguiente gráfico son los resultados de Accuracy para estos predictores sobre distintos valores de *W.*

La puntuación inicial tuvo siempre predicción sobre la misma clase (sin agregados), esto se debe a que el dataset no contaba con varias preguntas por lo que el modelo no aprendió a encontrar patrones en ese aspecto.



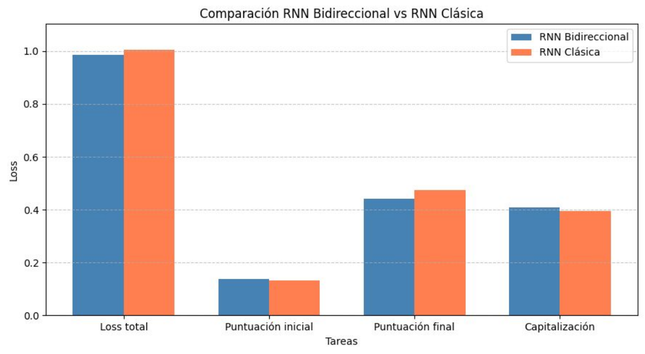
RNNs



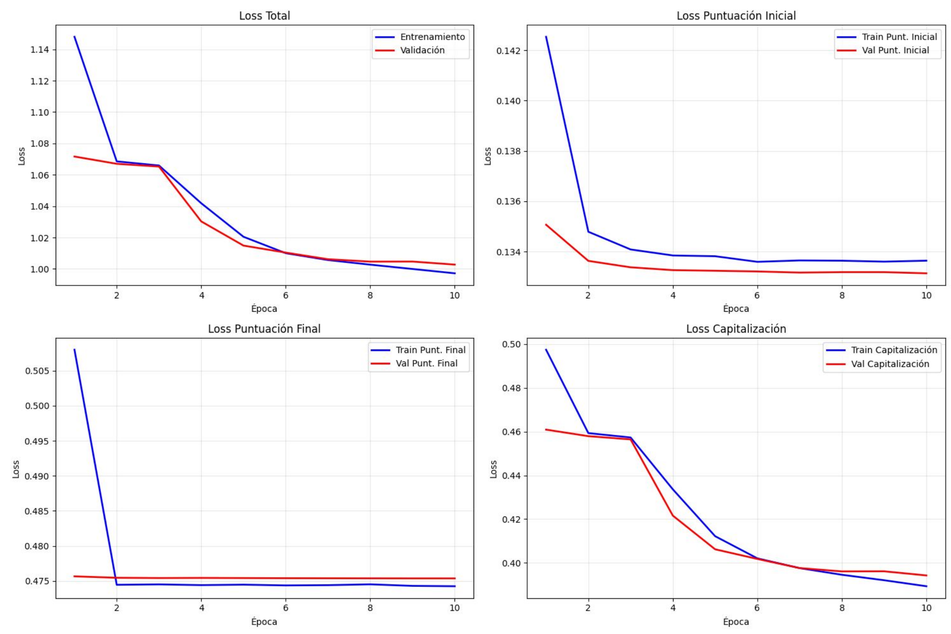
A. RNN unidireccional
Modelo con una capa de embeddings pre-entrenada (BERT multilingüe, congelada durante el entrenamiento) seguido de dos capas ocultas RNN unidireccionales, para luego dividir el modelo en 3 cabezas con RNNs unidireccionales para cada tarea de clasificación. El tamaño de los embeddings es de 768, utilizamos un estado oculto para todas las capas ocultas de 384. La salida es del output son 3 tuplas, una de 2 elementos y dos de 4.

B. RNN bidireccional
Arquitectura similar pero con 2 capas RNN bidireccionales antes de la división [1]**. Se usa la misma entrada de embeddings que en A, y la salida es la misma. Pero utilizamos 768 como tamaño para las capas bidireccionales, con lo que entre la capa forward y backward tienen un output de 1536 elementos. A diferencia del modelo A, se exploró también el ajuste fino (fine-tuning) del embedding preentrenado.**

Resultados con RNNs



Comparación de la Cross-Entropy con los datos de validación para los modelos A y B en cada una de las tareas.



Pérdida Cross-Entropy en el entrenamiento del modelo B

Conclusiones

Nuestros modelos no fueron capaces de aprender las tareas con nuestro entrenamiento y dataset. Tienden a responder clase 0 en casi todos los tokens. Sin Embargo, como la gran mayoría de los tokens no tiene ni capitalización ni puntuación, la Loss no resulta tan alta, puesto que la clase 0 se repite mucho.

Los datos utilizados para no fueron balanceados con respecto a las clases y casos a aprender, esto debe pesar bastante sobre la capacidad de aprendizaje de nuestros modelos.

Referencias:
1. Xu, Kaituo & Xie, Lei & Yao, Kaisheng. (2016). Investigating LSTM for punctuation prediction. 1-5. 10.1109/ISCSLP.2016.7918492.