PREDICCIÓN DE CAPITALIZACIÓN Y PUNTUACIÓN.

Falczuk N., Fernández Zaragoza M., Fiore J. I., Larrea J. I., Ombrosi U.

INTRODUCCIÓN

Objetivo: Diseñar y comparar tres modelos que predigan simultáneamente capitalización, puntuación inicial y puntuación final sobre texto plano a nivel de palabra.

Ejemplo: `hoy hace frío lloverá mañana` ⇒ `Hoy hace frío, ¿lloverá mañana?`

Modelos usados: Random Forest (clásico), Red Neuronal Recurrente (unidireccional y bidireccional).

ALGORITMO CLÁSICO: RANDOM FOREST

Ingeniería de atributos: Para entrenar el modelo Random Forest, se construyó un dataset con atributos a partir de texto tokenizado, incorporando información contextual, gramatical y semántica.

Atributos utilizados:

- **Embeddings reducidos:** de dimensión 2 a 256 (en potencias de 2) usando PCA sobre la primera capa de BERT y luego aplicando esa transformación sobre cada embedding.
- **Etiquetas gramaticales:** extraídas con `es_core_news` de spaCy (NOUN, VERB, PROPN, etc.). Usamos **one-hot encoding**.
- **Posición:** indicadores booleanos como `es_primer_token`, `es_último_token`.
- **Contexto:** embeddings de los tokens anteriores y siguientes, con ventana K.

token	token_id	TAG	inicio	fin	emb_0	prev_0	sig_0
un	10119	DET	1	1	-0.336	—	0.283
ray	27212	PROPN	1	0	0.283	-0.336	-0.235
##o	10133	UNK	0	1	-0.235	0.283	—

Ejemplo ilustrativo del dataset usado para entrenar RF.

Resultados:

Top 3 f1 macro para distintas combinaciones de atributos:

Capitalización

F1	K	Dim	Tags	Depth
0.67	1	2	Si	None
0.66	0	8	Si	None
0.66	0	2	Si	None

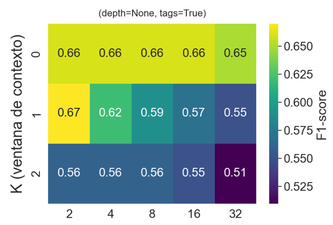
Puntuación inicial

F1	K	Dim	Tags	Depth
0.70	1	4	Si	None
0.70	1	16	No	None
0.70	1	16	Si	None

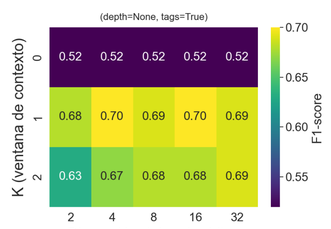
Puntuación final

F1	K	Dim	Tags	Depth
0.42	1	32	Si	None
0.41	1	16	Si	None
0.41	1	16	No	None

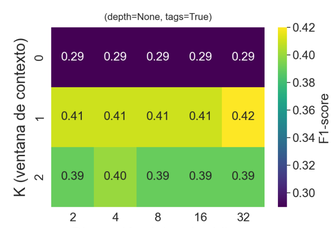
Heatmaps sobre F1 Macro (dimensión vs ventana K):



Capitalización

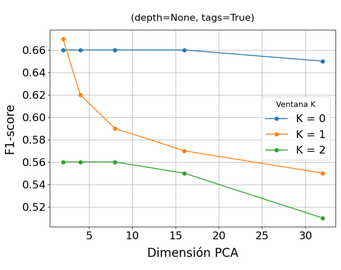


Puntuación inicial

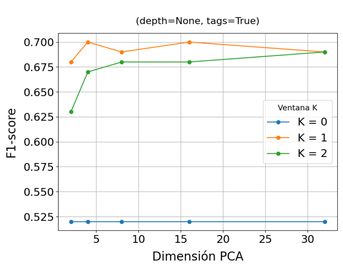


Puntuación final

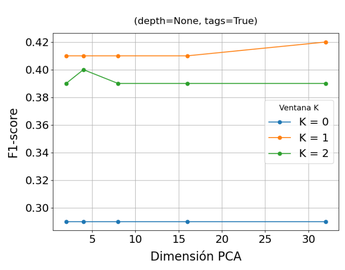
F1 Macro (eje Y) vs dimensión de PCA (eje X):



Capitalización



Puntuación inicial



Puntuación final

PRELIMINARES

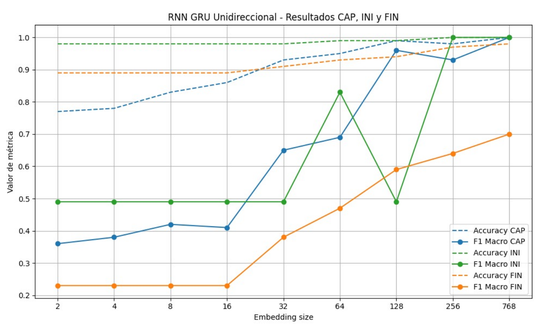
Datasets utilizados: EsCorpius (HuggingFace), CUNHA19 (HuggingFace), Borges Corpus (GitHub, karen-pal). Se tomaron subconjuntos filtrados, eliminando textos muy largos o con puntuaciones no deseadas (!, [], $, etc.).

Procesamiento: A partir de texto bien puntuado y capitalizado, armamos el dataset etiquetando texto plano. Usamos el modelo pre-entrenado BERT para tokenizar y obtener los embeddings de la primera capa, que agregan semántica y gramática en forma densa, ya que los `token_id` no aportan información útil como entrada directa.

Features: Para los tres modelos probamos usar diferentes dimensiones de los embeddings aplicando reducción con PCA. En el modelo Random Forest, además incorporamos atributos categóricos adicionales.

RNN ESTÁNDAR CON GRU

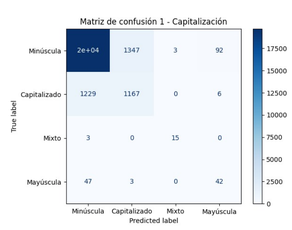
La arquitectura procesa una secuencia de embeddings con una GRU direccional, usando regularización dropout. La salida es utilizada por tres cabezas especializadas que predicen capitalización, puntuación inicial y final.

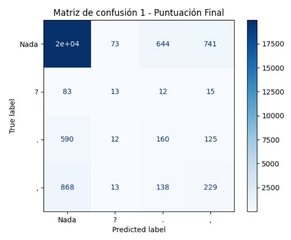


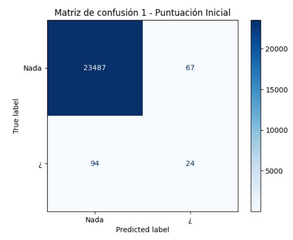
En un principio probamos, para muy pocos datos, cómo aprendía la red unidireccional para diferentes dimensiones de los embeddings, con el objetivo de ver si la red lograba aprender bien la tarea. En función de los resultados observados en F1 macro, decidimos continuar con dimensión igual a 256.

	Sin capa compartida	Con capa compartida
F1macro Capitalización	0.651	0.637
F1macro P. inicial	0.613	0.612
F1macro P. final	0.352	0.333

El entrenamiento se realizó alternando entre funciones de pérdida cross-entropy estándar y con pesos para mitigar el desbalance de clases usando un batch size de 32 instancias y 400 epochs. Además probamos incluir atención, pero la performance empeoró. También comparamos versiones con y sin capa compartida, como reportamos en la tabla. Como el rendimiento fue mejor sin la shared layer, optamos por el modelo más simple.







RED BIDIRECCIONAL CON GRU

En esta variante, utilizamos la misma arquitectura pero con una GRU que procesa la secuencia en ambos sentidos. La salida combinada se pasa por las cabezas especializadas. Para el entrenamiento se usaron 32 instancias por batch size y 200 epochs. En este caso volvimos a notar una mejor diferencia en capitalización y puntuación final para la red sin capa compartida, pero una leve mejor performance en puntuación inicial en la que sí presentaba capa compartida. En función de esto último, como nuestro mejor modelo era el de la red bidireccional sin capa, decidimos escalar los conjuntos de entrenamiento y de test sobre este modelo, entrenando con batch size = 8 y 300 epochs, y sin reducción de la dimensión en los embedding (768). Los resultados obtenidos se muestran sobre la tabla.

	Sin capa compartida (con train reescalado)	Con capa compartida
F1macro Capitalización	0.924	0.699
F1macro P. inicial	0.789	0.715
F1macro P. final	0.737	0.591

CONCLUSIONES

- Como era de esperar, el random forest no logra aprender relaciones complejas para predecir capitalización y puntuaciones. Quizás usar otro algoritmo clásico de ensambles como CatBoost que permite mezclar atributos categóricos y continuos sería una mejor opción.
- Las redes neuronales recurrentes unidireccionales no parecen haber mejorado respecto al algoritmo clàsico
- Los modelos de redes neuronales recurrentes bidireccionales mostraron buen desempeño en general. En particular, este soluciona el principal problema "¿", pues logra capturar información previa y posterior en la secuencia, y mejora el resultado en puntuación final.