Reconeixement d'entitats anomenades

Claudia Gallego Torralbo, Verónica Oñate Villagrasa

29 d'Octubre de 2025

Universitat Politècnica de Catalunya

Grau en Intel·ligència Artificial

Tractament de la Veu i el Diàleg

Índex

Introducció

En aquesta segona part de la pràctica, l'objectiu és aprendre a reconèixer entitats anomenades, NER. El reconeixement d'entitats anomenades consisteix a identificar i classificar entitats dins d'un text, en aquest cas sobre la reserva de vols, com ara ciutats, dates, tipus de viatge, entre altres categories específiques.

En aquesta pràctica, es seguiran passos similars als de la primera part, on es va treballar amb la classificació d'intencions. En primer lloc, s'analitzarà el conjunt de dades per conèixer les entitats a identificar. A continuació, es prepararan les dades per poder entrenar els models, i finalment, es dissenyarà l'arquitectura i es realitzarà l'entrenament dels models de reconeixement d'entitats anomenades. A través d'una sèrie d'exercicis pràctics, que s'expliquen més endevant, es podrà construir i avaluar un model capaç de identificar i classificar les entitats. Per fer-ho, s'ha dissenyat un conjunt d'experiments per analitzar l'impacte de diferents mètodes i arquitectures en el rendiment del model, incloent xarxes neuronals recurrents, convolucionals i transformers

Aquesta tasca de NER es presenta com una evolució natural de la tasca prèvia de classificació d'intencions, ja que ambdues són peces clau en la creació de sistemes NLP més sofisticats. Tant la classificació d'intencions com el reconeixement d'entitats anomenades es complementen i permeten als sistemes entendre no només la intenció general d'una conversa, sinó també les entitats específiques que poden ser crucials per a proporcionar respostes més precises i personalitzades en el context d'una conversa.

Experiments i Resultats

Com aquesta és la segona part de la pràctica, els primers exercicis era fer el mateix que a la primera part pero adaptant al nou problema.

Exercici 1

En aquest primer exercici es demana que es carreguin les bases de dades. Primer, es carreguen les particions de train i test i es defineix una mostra de validació separada a partir del train (les últimes 900 líneas del train). Es comprova que les mides de les particions siguien les correctes.

Exercici 2

Ara, de cada fila s'extreu la seqüència de tokens i la seqüència d'etiquetes corresponent en format BILOU, que són la primera i segona columna de la base de dades, per guardar-les a la partició corresponent. Abans de guardar-les es fa una neteja per eliminar caràcters sobrants i finalment imprimeix una mostra per veure que tot està Ben guardat.

Exercici 3

Es construeix un vocabulari amb els tokens d'entrenament i es transformen les oracions en índexs. Es registra la longitud original de cada oració per poder excloure el padding a l'avaluació. Tot seguit, s'aplica pad_sequences fins a la longitud màxima observada a train, garantint tensors de mida fixa coherents entre train/val/test.

Exercici 4

Per aquest exercici l'objectiu és convertir les diferents classes d'entitats en vectors one-hot. Per fer-ho, primer hem de saber quantes etiquetes diferents hi ha i assegurar-nos que enten les classes amb mateix nom però diferent etiqueta BILOU com a entitats diferents. Després, s'assigna un índex a cadascuna de les entitats úniques, incloent el <pad> en el índex del O. Finalment, es codifiquen les entitats en vectors one-hot.

Exercici 5

En aquest exercici s'ha de dissenyar l'arquitectura i entrenar el model per a la classificació d'intencions. Com es diu a l'exercici, s'ha uilitzat la mateixa arquitectura base que en la primera part de la pràctica que després serà millorada en l'exercici 6. S'ha implementat un model seqüencial amb Embedding, una BiLSTM, una capa densa intermèdia amb activació ReLU i una última capa densa amb sortida softmax amb tantes unitats com etiquetes.
Es compila amb Adam i loss categorical crossentropy. L'avaluació incorpora accuracy i F1 macro via classification report, descartant el <pad> i considerant només tokens reals. Finalment, es presenten mostres qualitatives comparant etiquetes reals i predites per validar el comportament del model.

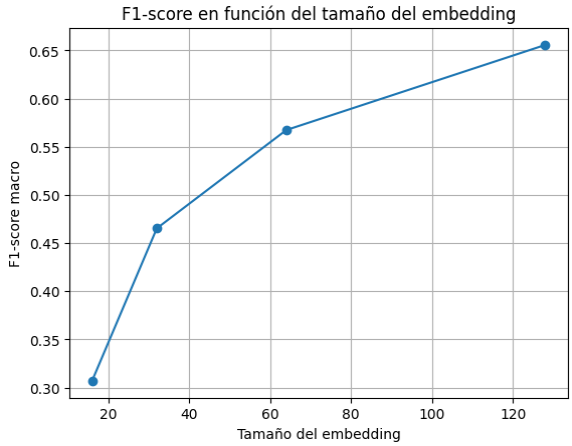
Exercici 6

6.1 Mida dels Embeddings

Explicació de l'experiment

L'objectiu és avaluar l'impacte de la dimensió de l'embedding sobre el model. Per fer-ho, es manté constants totes les altres decisions d'arquitectura i entrenament que hem utilitzat en l'exercici anterior. El model fa exactament el mateix que el que ja teniem implementat i es fa un bucle per a que es repeteix-hi l'entrenament i l'avaluació del model sobre la llista de *embedding_dim = [16, 32, 64, 128].*
L'avaluació és fa al conjunt de test amb accuracy i F1 macro, excloent els tokens de padding. També s'inspecciona el classification report del millor model segons F1 macro.

Comentari dels resultats



Els resultats milloren a mesura que l'embedding creix de 16 a 128, amb un lleuger augment de l'accuracy i una millora significativa de l'F1 macro (de 0,45 a 0,66) , especialment en classes minoritàries. El millor model amb embedding de 128 aconsegueix un F1 macro de 0,65, un micro F1 de 0,91 i un weighted F1 de 0,89, mostrant un bon rendiment global però amb desequilibris entre classes. Les etiquetes més comunes, com B-/I-fromloc.city_name i B-/I-toloc.city_name, obtenen un F1 alt (0,94–0,97), mentre que les etiquetes rares, com B-aircraft_code o B-fromloc.airport_name, tenen un F1 molt baix o nul.

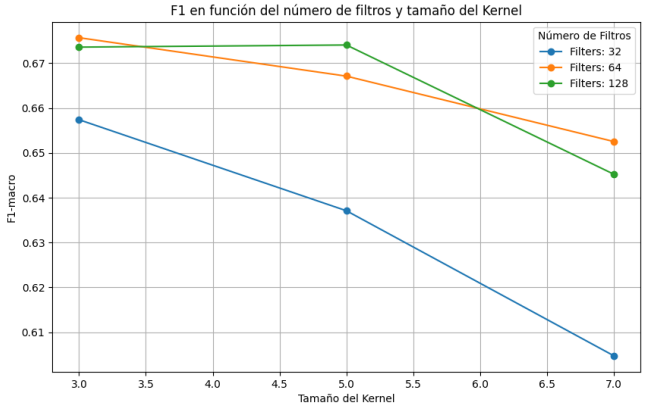
Aquesta millora es deu a que un embedding més gran captura millor el significat dels tokens, però l'F1 baix es manté a causa del desbalanceig del corpus.

6.2 Xarxes Convolucionals

Explicació de l'experiment

S'incorporen una cape Conv1D després de Embedding i la resta queda igual. Al igual que a l'anterior es fa un bucle, en aquest cas sobre els hiperparàmetres de mida filtres i mida Kernel. No s'aplica pooling per evitar perdre alineament token–etiqueta ja que el pooling redueix la mida temporal i obligaria a re-escalar/"desfer" la seqüència.

Comentari dels resultats



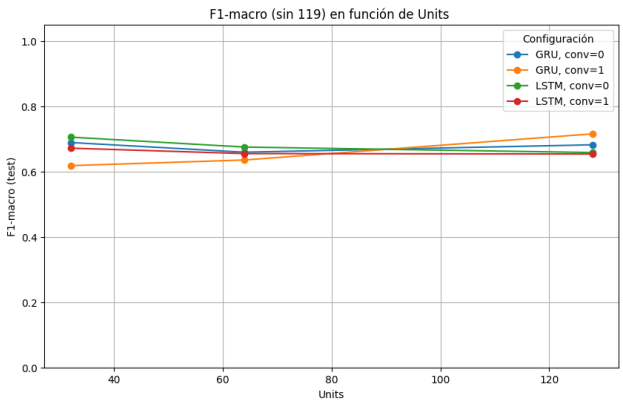
Els resultats mostren que 128 filtres i kernel de mida 3 van oferir els millors resultats, amb accuracy = 0,9844 i F1 macro = 0,67. S'observa que augmentar el nombre de filtres ajuda a millorar el rendiment, però quan s'augmenta la mida del kernel (5 o 7), l'accuracy i F1 macro disminueixen, indicant que mides de kernel més grans poden suavitzar massa les relacions locals i perdre detall important. Les entitats comunes com B-fromloc.city_name i B-toloc.city_name mantenen un F1 alt (0,96), però les menys freqüents com B-airport_code (F1 ≈ 0,43) tenen un recall més baix, la qual cosa afecta l'F1 macro global.

Xarxes Recurrents

Explicació de l'experiment

En aquest experiment s'han provar diferents configuracions de xarxes recurrents, comparant LSTM i GRU, de 32, 64 o 128 unitats per capa. També probant el model amb la capa convolucional y sense. L'objectiu era observar com cada paràmetre afecta el rendiment per escollir el millor.

Comentari dels resultats



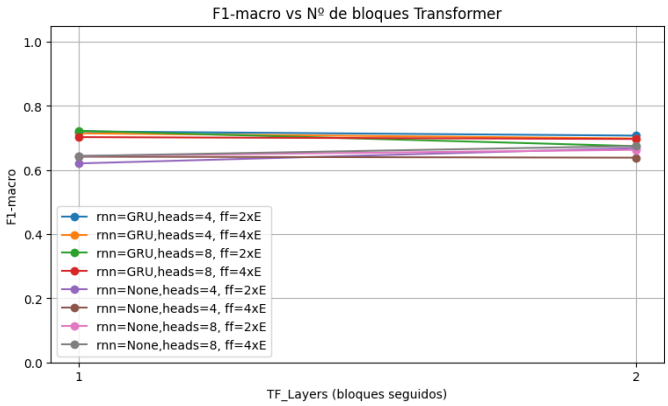
Els gràfics de resultats mostren que l'accuracy és força estable per a totes les configuracions, mantenint-se al voltant de 0,98. Tot i que la presència de la capa Conv1D no provoca canvis significatius en l'accuracy, sí que influeix en l'F1 macro. En general, la millor configuració va ser GRU amb 128 unitats i conv=1, obtenint F1 macro = 0,715. Això mostra que afegir una capa convolucional ajuda a millorar el rendiment en les classes menys representatives, encara que tampoc hi ha gaire diferència sense aquesta capa. Les diferències entre GRU i LSTM no són molt marcades, amb les dues configuracions produint resultats similars.

Transformer

Explicació de l'experiment

En aquest experiment s'han afegit blocs Transformer a l'inici del model abans de les capes recurrents. El model cambia la capa d'Embedding per una capa de Token i Positional Embedding i Transformer. S'han provat diverses configuracions variant el nombre heads (4 o 8), la dimensió del feed-forward (2×E o 4×E), el nombre de blocs Transformer (1 o 2) i amb Xarxes recurrents o sense.

Comentari dels resultats



Els resultats mostren l'F1 amb un clar benefici d'afegir GRU: els models sense RNN queden a 0.62–0.67, mentre que amb GRU pugen a 0.69–0.72. El millor resultat s'obté amb GRU + 1 bloc Transformer, 8 caps i FF=2×E, amb F1-macro = 0,722 i accuracy ≈ 0,98. Amb 4 caps i FF=2×E també s'aconsegueix un F1 alt (0,720). Afegir 2 blocs no sempre ajuda: en diverses combinacions el F1 es manté o baixa lleugerament. El model amb xarxes recurrents millora el rendiment perquè aquestes capes són capaces de capturar dependències temporals i seqüencials a llarg termini, com les relacions entre les paraules d'una oració. Això ajuda a millorar el recall i el F1 macro per a les classes menys freqüents, que són més difícils de detectar només amb Transformers. El classification report del millor model reflecteix F1 alt en entitats freqüents (p. ex. fromloc/toloc.city_name ≈0,97) i millores raonables en algunes categories mitjanes (fare_basis_code ≈0,81), però persisteixen etiquetes rares amb recall baix (aircraft_code, I-flight_mod).

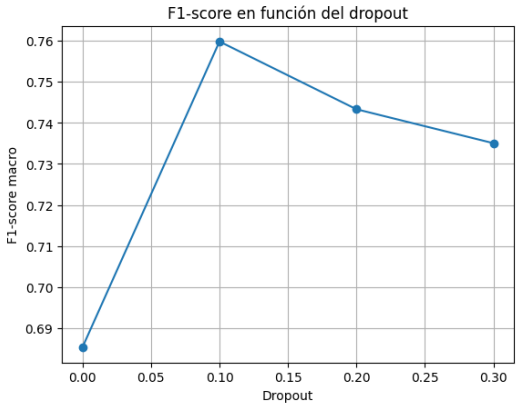
Regularització

Explicació de l'experiment

En aquest experiment s'ha provat l'ús de dropout per regularitzar el model i evitar el sobreajustament. S'ha aplicat dropout en dues parts del model: després de l'embedding, així com després de les capes feedforward de ReLu. El dropout es va variar entre 0% i 30%, i es van mesurar els efectes sobre l'accuracy i l'F1 macro. S'utilitza el model amb els paràmetres que millors resultats han donat en l'exercici anterior, un model Transformer amb 1 bloc, 8 caps d'atenció i feed-forward de 2×E.

Comentari dels resultats

Els resultats mostren que l'ús de dropout millora clarament l'F1 macro en un rang entre 0% i 10% de dropout, on el model aconsegueix un F1 macro màxim de 0,759 a 0,1 de dropout. Aquest valor es manté estable a 0,74–0,76 en configuracions entre 0,1 i 0,2 de dropout. No obstant això, quan el dropout supera el 20%, el rendiment comença a disminuir, amb un F1 macro de 0,735 a 0,3 de dropout, indicant que un dropout excessiu pot fer que el model perdi capacitat d'aprenentatge.



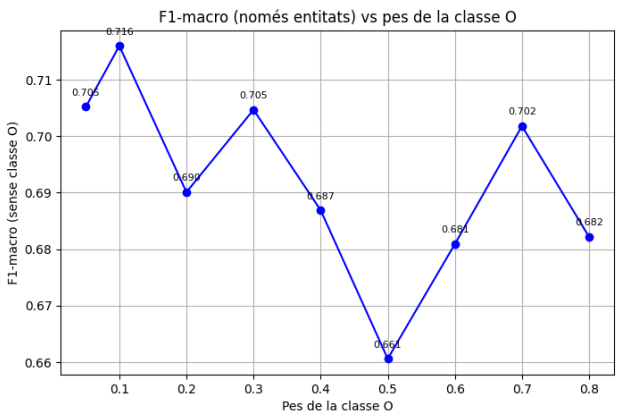
Balancejat de les classes

Explicació de l'experiment

L'objectiu d'aquest experiment és corregir el desbalanceig de classes del corpus, on la classe O domina amb més del 90 % de les etiquetes. Per evitar que el model aprengui a predir gairebé sempre O, s'ha aplicat una pèrdua ponderada (loss amb pesos per classe) que dona més importància a les entitats minoritàries i redueix el pes d'O.
S'han provat diferents valors per al pes de la classe O, entre 0,8 i 0,05, mantenint pesos inversos per a la resta. Després, s'ha entrenat el mateix model que hem utilitzat en l'anterior amb dropout de 0.1.

Comentari dels resultats

Els resultats mostren que reduir el pes d'O millora l'F1-macro de manera clara fins a un cert punt. Els millors valors s'obtenen amb pes(O)=0,1–0,3, on l'F1-macro arriba a 0,70–0,72.
Quan el pes d'O és massa alt (0,8–0,6), el model esbiaixa fortament cap a aquesta classe i les entitats minoritàries tenen un recall molt baix. En canvi, amb pesos massa baixos (0,05), el model perd estabilitat i baixa lleugerament la precisió.



Quan s'aplica ponderació per classes, pot semblar que hauria de millorar sempre el rendiment global, però no necessàriament passa així. En aquest cas, el model sense desbalanceig arriba a un F1 = 0,76, lleugerament superior al millor resultat amb pesos (≈0,72).

Això pot passar per diversos motius:

- Soroll en classes rares: les etiquetes minoritàries tenen molt poques mostres i, en donar-los un pes alt, el model pot sobreadaptar-se a patrons poc fiables, empitjorant el rendiment global.
- Més errors en classes freqüents: en reduir el pes de la classe O, el model deixa d'optimitzar tant la precisió general, provocant petites pèrdues en etiquetes comunes que, al final, també afecten l'F1 macro.

- Equilibri artificial: en un corpus tan desbalancejat (90 % O), un pes mal ajustat pot trencar l'equilibri natural del model i fer-lo menys consistent.

En resum, tot i que el balanceig ajuda teòricament a millorar el recall de classes rares, en aquest cas el model sense pesos pot haver aprofitat millor la regularitat de les dades i entrenat de manera més estable, donant un F1 lleugerament superior.

Conclusions

Els experiments realitzats mostren que l'elecció dels models i dels hiperparàmetres té un impacte important en el rendiment del reconeixement d'entitats anomenades (NER). En primer lloc, s'ha observat que augmentar la mida de l'embedding millora els resultats, especialment en les classes minoritàries. Les xarxes convolucionals (Conv1D) van contribuir a millorar l'F1 macro, però l'augment de la mida del kernel va reduir l'eficàcia. Les xarxes recurrents (GRU) amb capa convolucional van aconseguir els millors resultats per a les classes menys representatives, mentre que l'ús de Transformer va millorar encara més els resultats quan es va combinar amb xarxes recurrents.

En quant a la regularització, l'ús de dropout va millorar el rendiment fins a cert punt, amb un valor òptim entre el 0% i el 10%. En quant al desbalanceig de les classes, es va veure que ponderar les classes pot ajudar a millorar el recall de les classes rares, tot i que el model sense ponderació va obtenir un F1 lleugerament superior.

En general, els resultats indiquen que una combinació de tècniques com les xarxes recurrents, Transformer, la regularització i el balanceig de les classes poden millorar el rendiment del model en tasques de reconeixement d'entitats anomenades. Tanmateix, és essencial ajustar els hiperparàmetres segons les característiques del conjunt de dades.