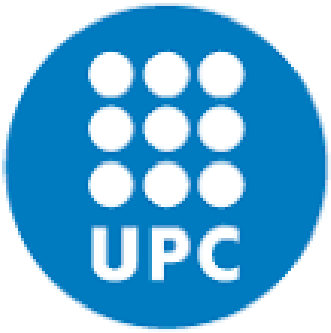
Universitat Politècnica de Catalunya
Grau en Intel·ligència Artificial

Tractament de la veu i el dialeg

Pràctica 2.1: Classificació d'intencions

Claudia Gallego, Verónica Oñate

13 de novembre de 2025



Índex

1 Introducció

Aquesta pràctica té com a objectiu aprendre a identificar la intenció de l'usuari a partir de les seves interaccions amb un xatbot. Per a això, s'utilitza una base de dades composta per oracions en espanyol, cadascuna etiquetada amb una de les 19 intencions diferents.

L'estructura del projecte s'ha organitzat en diversos exercicis, cadascun enfocat en una fase específica del procés de classificació de les intencions. Els tres primers exercicis estan dedicats a l'exploració i el processament de les dades. Els dos exercicis restants es centren en el disseny i entrenament d'un model de classificació.

El document entregat és un notebook de Jupyter, estructurat per seccions segons els diferents exercicis i les tècniques aplicades en cada etapa.

2 Processament de les dades

2.1 Exercici 1: Conversió de les oracions a seqüències d'IDs

En aquest primer exercici, l'objectiu ha estat convertir les oracions de la partició d'entrenament (train_sentences) en seqüències d'IDs mitjançant un tokenitzador.

Primer, s'ha creat el tokenitzador amb les 500 paraules més freqüents i s'ha ajustat al conjunt d'entrenament per construir el vocabulari, assignant un ID a cada paraula.

A continuació, s'han convertit les oracions en seqüències d'IDs mitjançant la funció `texts_to_sequences` i s'ha aplicat el padding per assegurar que totes les seqüències tinguin la mateixa longitud, facilitant així el seu ús en el model de classificació.

2.2 Exercici 2: Millora del padding i codificació de les etiquetes

En aquest segon exercici, s'ha millorat el procés de padding de les seqüències d'IDs i també s'han codificat les etiquetes per al model.

Primer, s'ha modificat l'aplicació del padding. S'ha utilitzat l'argument `maxlen` per establir una longitud màxima de seqüències, i l'argument `padding='post'` ha permès afegir els zeros al final de les seqüències més curtes, evitant així pèrdues d'informació.

A continuació, s'ha codificat les etiquetes de les oracions mitjançant una codificació numèrica. Primer, s'ha utilitzat un `LabelEncoder` per convertir les etiquetes originals en valors enters, assignant un identificador únic a cada classe. A continuació, s'ha aplicat la codificació one-hot amb la funció `to_categorical` per convertir els valors numèrics en vectors binaris, on cada etiqueta es representa com un vector amb un 1 a la posició de la classe corresponent i 0s a la resta de posicions.

Aquest procés assegura que tant les oracions com les seves etiquetes estiguin en un format adequat per a l'entrenament d'un model de classificació.

2.3 Exercici 3: Aplicació als conjunts de validació i test

En aquest tercer exercici, es realitzen els mateixos passos que en els dos exercicis anteriors, però ara aplicats als conjunts de validació (val) i de test (test). Això permet garantir que tant les dades de validació com les de test siguin transformades de la mateixa manera que les dades d'entrenament, assegurant coherència i compatibilitat amb el model.

3 Disseny del model i entrenament

3.1 Exercici 4: Creació del model segons especificacions

Aquest exercici tracta de la creació d'un model de xarxa neuronal per a la classificació de textos seguint les especificacions donades. El model inclou:

1. **Capa d'Embedding** per transformar paraules en vectors d'alta dimensió.
2. **Capa de MaxPooling** per reduir la dimensió de les dades i extreure les característiques més importants.
3. **Capa Flatten** per convertir les dades de 3D a 2D.
4. **Capa Densa** amb activació ReLU per aprendre relacions no lineals.
5. **Capa de sortida** amb activació Softmax per predir les probabilitats de cada classe.

Els resultats d'aquest primer model són satisfactoris, ja que les mètriques `accuracy` i `F-1 score` mostren uns valors de 0.93 i 0.921 respectivament. La seguent matriu mostra la distribució de les prediccions del model per a cada classe.

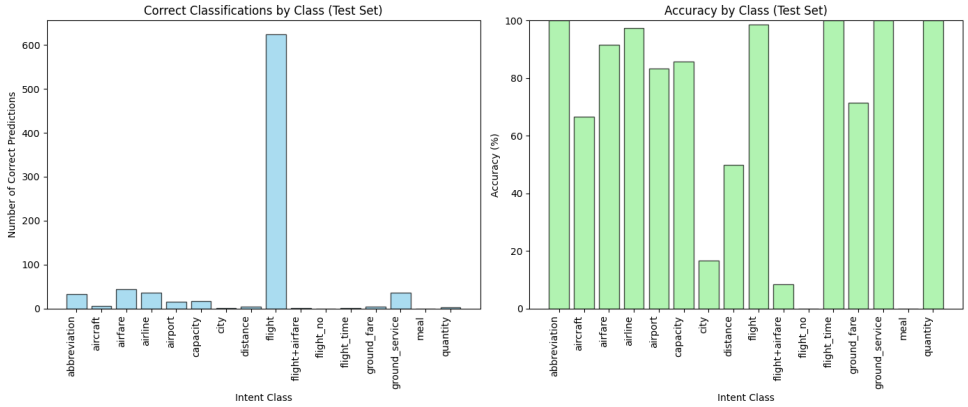


Figura 1: Gràfiques de prediccions per classes

Tot i que els resultats han estat satisfactoris, es pot observar (Figura 1) que la classe `flight` està sobredimensionada en comparació amb la resta. Com a conseqüència, classes com `flight-no` i `meal` mostren un `accuracy` molt proper a 0, i seria necessari aplicar tècniques de balanceig de dades o ajustar els hiperparàmetres del model.

3.2 Exercici 5:

En l'Exercici 5, es va modificar el model anterior per millorar les seves mètriques d'avaluació a través de diversos ajustos. Es van realitzar canvis en el preprocessament, com la modificació del Tokenizer per ajustar la mida del vocabulari i afegir tècniques com la lemmatització. També es van provar diferents mides d'**Embeddings** per analitzar el seu impacte en el rendiment. A més, es van afegir capes convolucionals i recurrents (LSTM, GRU) per explorar com aquestes arquitectures afecten l'accuracitat. Per evitar el sobreajustament, es va incorporar la regularització mitjançant **Dropout**, i finalment, es va aplicar el balanceig de les classes utilitzant els pesos de classe per ajustar la funció de pèrdua, donada la distribució desequilibrada de les classes en el conjunt de dades.

3.2.1 Mida del vocabulari

L'objectiu d'aquesta modificació era estudiar com la mida del vocabulari influeix en el rendiment del model. Per a això, s'han mantingut constants totes les altres decisions d'arquitectura i entrenament utilitzades en l'exercici 4. S'han provat diferents mides de vocabulari per al tokenizer, incloent-hi els valors de 300, 500 i 800 paraules. L'avaluació s'ha fet en el conjunt de test amb les mètriques d'accuracy i F1 macro.

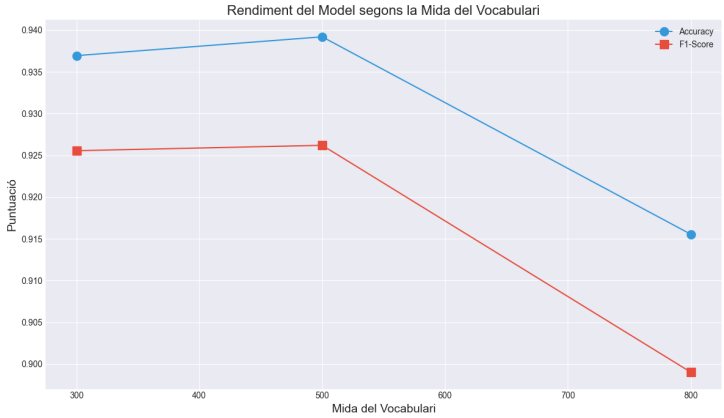


Figura 2: Gràfica del rendiment del model

Els resultats obtinguts mostren que augmentar la mida del vocabulari no sempre es tradueix en una millora del rendiment del model. Inicialment, s'observa que entre 300 i 500 paraules, tant l'accuracy com el F1-score milloren lleugerament, indicant que un vocabulari més gran permet al model capturar millor la variabilitat del text. No obstant això, quan la mida del vocabulari s'incrementa a 800 paraules, tant l'accuracy com el F1-score experimenten una caiguda significativa. Això suggereix que un vocabulari massa gran pot introduir més soroll i paraules poc freqüents, dificultant la tasca del model.

En general, els millors resultats s'han obtingut amb un vocabulari de mida moderada, al voltant de les 500 paraules.

3.2.2 Aplicació de Lematització i Stemming

L'objectiu d'aquesta modificació era observar com les tècniques de lematització i stemming afecten l'entrenament del model i el seu rendiment. La lematització consisteix en reduir les paraules a la seva forma base, tenint en compte el seu context, mentre que el stemming elimina les terminacions morfològiques per reduir les paraules a les seves arrels. Ambdós processos van ser aplicats als textos abans de ser introduïts al model base,

Igual que en l'exercici anterior, no es va aplicar cap altra modificació al model per tal d'observar la millora relativa de manera més objectiva.

Els resultats mostren clarament que aplicar stemming millora el rendiment del model respecte al model base en termes d'accuracy i F1-score. D'altra banda, l'aplicació de lematització no millora els resultats i podria ser degut a que aquesta tècnica funciona millor quan es combina amb altres tècniques de preprocessament.

3.2.3 Mida dels Embeddings

L'objectiu d'aquesta modificació era observar com la dimensió de l'embedding afecta l'entrenament i accuracy del model. Per fer-ho, s'han mantingut constants totes les altres

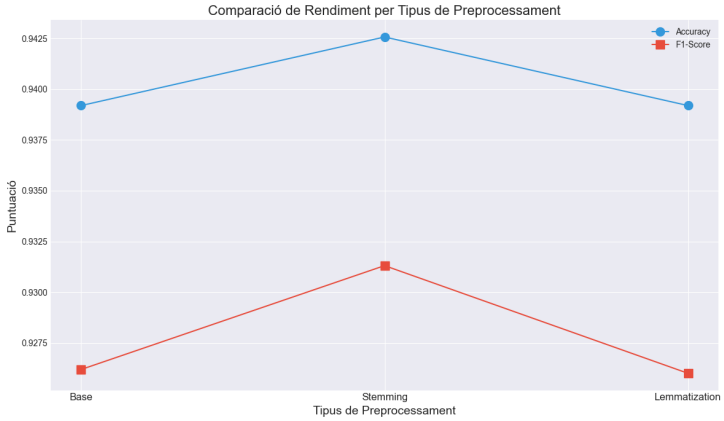


Figura 3: Gràfica del rendiment del model

decisions d'arquitectura i entrenament que hem utilitzat en l'exercici 4, al igual que s'ha fet en els casos anteriors. Les mides dels embeddings provats han sigut 50, 75, 100, 150 i 200.

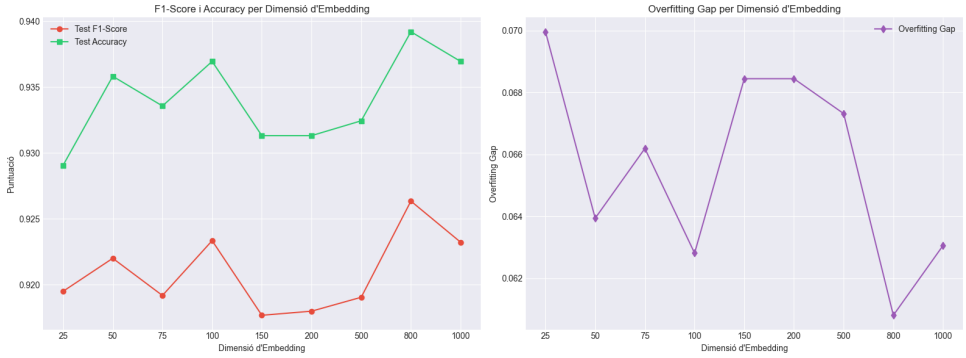


Figura 4: Gràfica del rendiment del model

La dimensió d'embeddings òptima és 100, ja que és on s'obtenen els millors valors de accuracy i F1-score. mentre que dimensions més grans no mostren millores significatives i fins i tot poden reduir el rendiment.

Les dimensions més baixes, com 25 i 50, presenten un rendiment inferior, ja que no capturen prou informació per al model, la qual cosa es tradueix en una menor precisió i F1-score.

3.2.4 Aplicació de capes convolucionals

L'objectiu d'aquesta modificació era explorar com l'ús de capes convolucionals afecta el rendiment del model en tasques de classificació de textos. Per a això, s'han afegit capes convolucionals de diferents configuracions (nombre de filtres i mida de kernel) a l'arquitectura del model, mantenint constants les altres decisions d'entrenament.

Les proves inclouen diverses configuracions de filtres (32, 64 i 128) i diferents mides de kernel (3, 5, 7).

Els resultats mostren que, en general, a mesura que s'augmenta el nombre de filtres (F), el model millora el rendiment en accuracy, arribant al seu màxim en la configuració F32 K3 max, amb un accuracy de 0.955, sent aquesta la millor configuració d'aquest experiment.

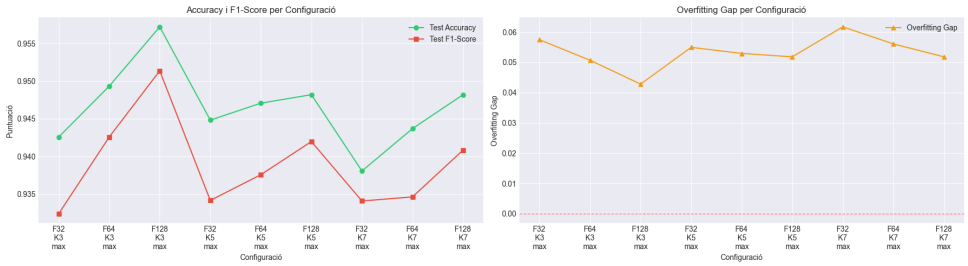


Figura 5: Gràfica del rendiment del model

Pel que fa a la mida del kernel (K), els resultats indiquen que, a mesura que la mida del kernel augmenta (de K3 a K5 i K7), el gap de sobreajustament també augmenta, especialment en les configuracions F32 K5 max i F64 K7 max. Això suggereix que les configuracions amb kernels més grans poden conduir a un sobreajustament més elevat, indicant que una mida de kernel més petita pot ser més efectiva per evitar aquest problema.

3.2.5 Aplicació de capes recurrents

L'objectiu d'aquesta modificació era analitzar l'impacte de les capes recurrents en el rendiment del model per a tasques de classificació de textos, on les dependències a llarg termini poden ser importants. Per a això, s'han afegit capes recurrents (com LSTM o GRU) a l'arquitectura del model, mantenint constants les altres decisions d'entrenament. Les capes recurrents són eficients per modelar seqüències de dades, com textos, capturant les relacions entre les paraules que poden estar separades en el temps. S'ha avaluat el rendiment del model en el conjunt de test utilitzant les mètriques d'accuracy i F1 macro, comparant els resultats obtinguts amb i sense l'aplicació de capes recurrents. També s'ha explorat l'efecte de variar el nombre de unitats recurrents per analitzar com influeix en la capacitat del model per aprendre seqüències complexes.

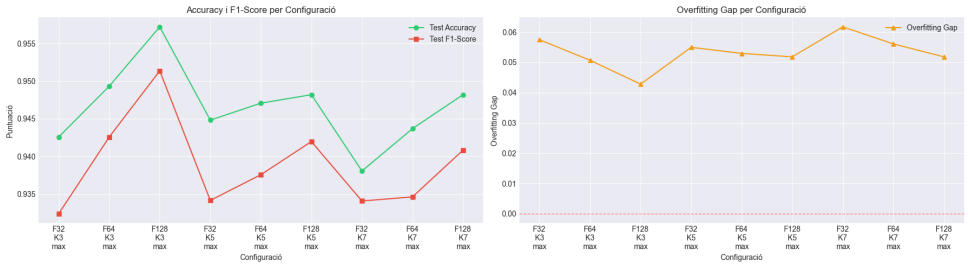


Figura 6: Gràfica del rendiment del model

Els resultats mostren que la configuració LSTM només té un bon rendiment quan les capes recurrents són bidireccionals (Bi), cosa que no passa en el cas de GRU, on ambdues configuracions (unidireccionals (Uni) i bidireccionals (Bi)) presenten un bon rendiment.

En el cas de GRU, augmentar el nombre de unitats incrementa directament els valors de les mètriques de validació, és a dir, augmenta el rendiment del model. Això no passa amb les capes recurrents LSTM, que obtenen resultats lleugerament inferiors al incrementar el nombre de unitats.

En general, el model que utilitza la capa recurrent GRU bidireccional amb 128 unitats, que és el valor provat més alt, mostra els millors resultats. Això podria indicar que provar

nombres més elevats d'unitats podria continuar augmentant el rendiment del model.

3.2.6 Regularitzar

L'objectiu d'aquesta modificació era explorar com l'ús de tècniques de regularització, com el dropout, pot ajudar a millorar la generalització del model i reduir el sobreajustament (overfitting). Per a això, s'ha aplicat la tècnica de dropout en les capes del model, amb l'objectiu de forçar el model a aprendre representacions més robustes i menys dependents d'un subconjunt específic de les característiques d'entrada. El dropout es va aplicar amb diverses taxes (0.2, 0.5, 0.7) per observar com aquesta variació influeix en el rendiment. L'avaluació s'ha realitzat sobre el conjunt de test, utilitzant les mètriques d'accuracy i F1 macro, comparant els resultats obtinguts amb i sense l'ús de dropout. Aquest experiment ha permès valorar l'efecte de la regularització sobre la capacitat del model per generalitzar als nous exemples i reduir el sobreajustament.

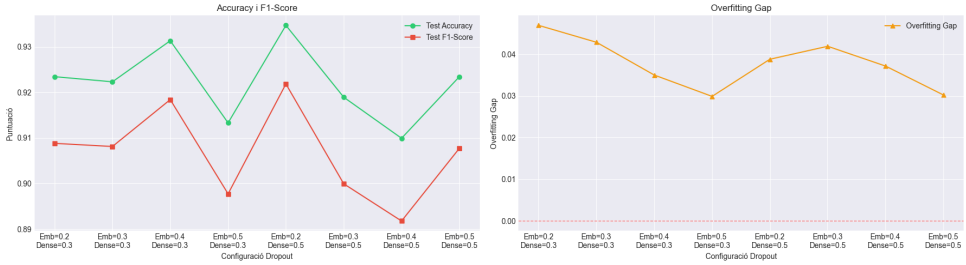


Figura 7: Gràfica del rendiment del model

Els resultats mostren que la configuració de dropout amb Emb=0.2 i Dense=0.5 presenta el millor rendiment en accuracy, arribant a una puntuació màxima de 0.93, mentre que el F1-score és lleugerament inferior. Aquesta configuració és la millor pel que fa a la generalització del model.

En augmentar el dropout, el model millora el rendiment fins a cert punt, però després comença a decaure, indicant que hi ha un límit en l'efectivitat de la regularització. Quan el dropout és massa alt, el model pot començar a perdre informació important.

El gap de sobreajustament (Overfitting Gap) mostra una tendència on l'augment del dropout en els embeddings ajuda a reduir el sobreajustament, demostrant que la regularització és efectiva per controlar aquest problema en el model.

3.2.7 Balanceig de les classes

L'objectiu d'aquesta modificació era abordar el problema de les classes desequilibrades, que pot afectar negativament el rendiment del model, especialment en tasques de classificació. Quan les classes no estan igualament distribuïdes, el model tendeix a predir en excés la classe majoritària, passant per alt la classe minoritària. Per resoldre aquest problema, s'han aplicat tècniques de balanceig de les classes, com submostreig de la classe majoritària (undersampling), sobresampling de la classe minoritària (oversampling) o pesos ajustats en la funció de pèrdua. Aquestes tècniques ajuden el model a aprendre a identificar amb més precisió les instàncies de la classe minoritària, millorant així les mètriques com l'accuracy, F1 score i AUC en el conjunt de test. L'impacte d'aquestes tècniques es va avaluar comparant els resultats obtinguts abans i després de l'aplicació del balanceig.

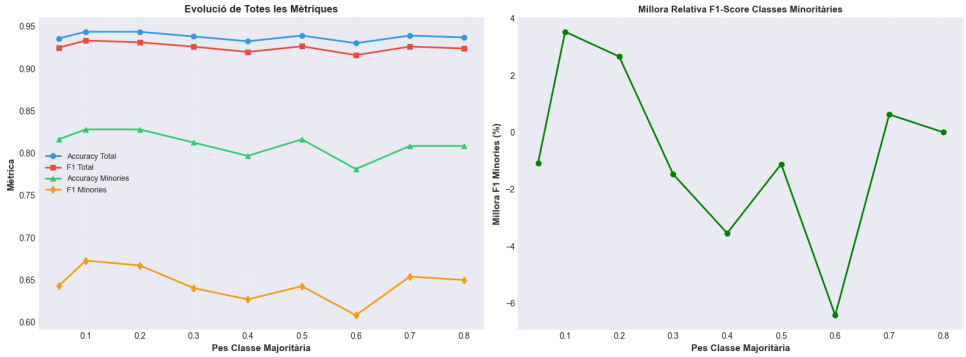


Figura 8: Gràfica del rendiment del model

Els resultats mostren que reduir el pes de la classe majoritària (O) millora l'F1-macro de manera clara fins a un cert punt. Els millors valors es van obtenir amb pesos d'O entre 0,1 i 0,3, on l'F1-macro arriba a 0,93-0,94. Quan el pes d'O és massa alt (entre 0,8 i 0,6), el model esbiaixa fortament cap a aquesta classe, i les entitats minoritàries obtenen un recall més baix. En canvi, amb pesos més baixos (per exemple, 0,05), el model perd estabilitat i disminueix lleugerament la precisió.