Facultat d'Informàtica de Barcelona

FIB UPC
Processament del Llenguatge Humà

Pràctica 1: Identificació d'Idioma

Alumnes :
Granja Bayot, Jordi
Jerez Cubero, Alberto

Tutors :
Medina, Salvador
Turmó, Jordi

March 4, 2024

Contents

1 Introducció

El present informe constitueix la justificació de la solució proposada pel problema plantejat. Aquest és, programar un identificador d'idiomes per les següents llengües europees: anglès, castellà, neerlandès, alemany, italià i francès.

Entre els objectius, destaca la introducció de conceptes i eines de *Natural Language Processing*, així com l'anàlisi de quins tractaments dels textos s'adeqüen més a la tasca de distingir certs idiomes. En aquest sentit, farem servir com a dades el 'wortschatz leipzig corpora', conformat per 30k i 10k frases de cadascuna de les llengües, pel *training set* i el *test set*, respectivament. D'aquesta manera, podrem entrenar un model de ML de classificació supervisat. Concretament, tractarem un classificador generatiu probabilístic, en quant que predirà la classe que més probablement podria haver generat cert text. Aquest és, el classificador multinomial naive de Bayes.

2 Preprocessament de les Dades

En primera instància, el procés de preprocessat de les dades és essencial, en quant que s'han de fixar certes regles per poder entrenar el model correctament. I és que, principalment, no volem que hi hagi cap distinció explícita entre les dades dels diferents idiomes que s'esdevingui del format d'aquestes. En aquest sentit, incidirem en quatre tractaments:

- **Eliminació de dígits:** els caràcters numèrics estan presents en totes les llengües, de manera que no són útils en la tasca de classificació.
- **Concatenació de frases:** amb el fi de tractar una única *string*, serà d'utilitat intercanviar els salts de línia per espais dobles. D'aquesta manera, fent servir trigrams de caràcters, evitarem formar tri-grams amb caràcters de frases amb contextos diferents.
- **Conversió de majúscules:** transformarem els textos a paraules amb només minúscules. Així, previndrem possibles errors en l'ús de majúscules als *raw texts*, a més d'obtenir tri-grams amb freqüències més distingibles.
- **Unificació d'espais:** substituirem tots els espais en blanc continus per un de sol, de manera que cap fitxer de dades presentarà distincions en el format del text.

Per dur a terme aquest procés, han estat útils les expressions regulars que Python proporciona. Així, hem pogut intercanviar patrons de caràcters dels textos, segons els tractaments anteriorment descrits.

```
1 def reduce_whitespaces(text: str) -> str:
2     #Return: string with only one whitespace between
         characters
3     return re.sub(r'[ ]+', ' ', text)
```

Listing 1: Exemple de substitució d'expressions regulars

3 Entrenament del Classificador Naive de Bayes

En aquest apartat entrem a fons en l'entrenament del model de detecció d'idioma. En el nostre cas, hem considerat oportú programar el classificador en una classe `LanguageIdentifier`, que ens ofereix organització del codi i encapsulació de funcionalitats: entrenament, predicció, etc. Així, de cara a la utilització del model, la inicialització d'una instància del classificador implicarà el seu entrenament, de manera que l'objecte resultant podrà predir múltiples vegades, donant versatilitat al model.

Model de predicció

Per fer les prediccions, busquem l'idioma que més probablement podria haver generat cert text, en el nostre cas, seran frases.

$$l_i = \arg\max_{l \in L} P(\hat{d}|l) \underbrace{=}_{\text{Bayes' theorem}} \arg\max_{l \in L} \frac{P(\hat{d}|l) \cdot P(l)}{P(\hat{d})} \tag{3.1.1}$$

Podem simplificar Eq. 3.1.1 deixant anar el denominador $P(\hat{d})$. Això és possible perquè es calcularà $P(\hat{d}|l)$ per a cada llenguatge possible, però $P(\hat{d})$ no canvia per a cada classe, ja que el còrpora és el mateix.

$$l_i = \arg\max_{l \in L} P(\hat{d}|l) = \arg\max_{l \in L} P(\hat{d}|l) \cdot P(l) \tag{3.1.2}$$

Sense pèrdua de generalització, podem representar el document $\hat{d}$ com un conjunt d'atributs que, en el nostre cas, seran trigrames $f_1; f_2; ...; f_n$.

$$l_i = \arg\max_{l \in L} P(f_1, f_2, ..., f_n|l) \cdot P(l) \tag{3.1.3}$$

En aquest punt, estimar la probabilitat condicional de cada combinació de trigrames resulta molt costós, de manera que aplicar l'assumpció Naive de Bayes simplifica molt el model. Aquesta és, la independència entre les probabilitats de cada trigrama, de manera que es poden multiplicar com segueix:

$$P(f_1, f_2, ..., f_n|l) = P(f_1|l) \cdot P(f_2|l) \cdot ... \cdot P(f_n|l) \tag{3.1.4}$$

Per tant, l'equació final del model de detecció d'idioma és:

$$l_i = \arg\max_{l \in L} P(l) \prod_{i=1}^{n} P(f_i|l) \tag{3.1.5}$$

No obstant, per evitar *underflow* normalment els càlculs es fan en l'espai logarítmic, quedant l'equació:

$$l_i = \arg\max_{l \in L} \log P(l) + \sum_{i=1}^{n} \log P(f_i|l) \tag{3.1.6}$$

En aquest punt, no podem determinar exactament les probabilitats de l'equació, de manera que a partir del còrpora les haurem d'estimar. Pel que fa a la $P(l)$, l'estimació és la mateixa per a tots els idiomes, ja que disposem del mateix nombre de documents per cada idioma. Així, podem reduir l'equació de predicció:

$$l_i = \arg\max_{l \in L} \sum_{i=1}^{n} \log P(f_i|l) \tag{3.1.7}$$

D'aquesta manera, només ens cal l'estimació de la probabilitat de què un trigrama e_j hagi estat generat per un llenguatge l ($\hat{P}(f_i|l)$). En el nostre cas, farem servir *Maximum Likelihood Estimation*, provant dos tipus de *smoothing*: Lidstone i *Absolute Discounting*. Bàsicament, es tracta de reservar massa de probabilitat a trigrames no vists al còrpora, evitant així probabilitats nul·les.

$$\hat{P}_{LID}(e_j) = \frac{C_d(e_j) + \lambda}{N_d + \lambda B} \quad B\text{: trigrames potencialment observables.} \tag{3.2.1}$$

$$\hat{P}_{ABS}(e_j) = \begin{cases} \frac{C_d(e_j) - \delta}{N_d}, & \text{si } C_d(e_j) > 0 \\ \frac{(B - N_0)\delta / N_0}{N_d}, & \text{altrament} \end{cases} \quad N_0\text{: nombre de trigrames no vists a } d. \tag{3.2.2}$$

Tal com il·lustren els dos mètodes d'estimació, haurem de fixar el valor de certs paràmetres: B i λ o δ per Lidstone i Absolute Discounting, respectivament, que fan referència a quanta massa de probabilitat reservem als trigrames no observats. Pel que fa a B, considerarem que el nostre vocabulari és el nombre total de trigrames que es poden formar. En el nostre cas, hem fixat $B = \text{PR}(24, 3) = 24^3$; tot i que el nombre de caràcters possibles en el còrpora podria ser un nombre més elevat, hem considerat que hi ha combinacions que no es poden donar en cap idioma (e.g. 'ñqw'). D'altra banda, segons l'*State of Art* hem assumit $\lambda = 0.5$ i per definir δ, hem fet servir el mètode de Ney et al. (1994) [1], que estableix δ com a funció de N_1 i N_2, el nombre de trigrames que tenen un recompte d'1 i un recompte de 2 al còrpora, respectivament:

$$\delta = \frac{N_1}{N_1 + 2N_2} \tag{3.2.3}$$

No obstant, els paràmetres de *smoothing* es poden canviar per l'usuari en la crida del mètode de predicció `identify_language`, sempre i quan estiguin dins del rang (0, 1). De fet, s'ha provat a partir de l'experiment que descriu el fitxer 'best_alpha.py' que, per un remostreig de les dades de *train*, la **millor** λ en aquest context és 0,9.

En aquest punt, identifiquem com a entrenament del model el procés d'obtenir tota la informació de trigrames del nostre còrpora, preprocessat segons s'indica a la secció 2. Això és, el comptatge de cada trigrama (mètode `get_count`) i el nombre total d'ocurrències de trigrames (N_d), que s'obtenen amb el mètode auxiliar `get_trigrams` per a cada idioma.

4 Predicció del Classificador Naive de Bayes

En aquest apartat entrem a fons en el procés de predicció del model de detecció d'idioma. Aquesta funcionalitat ve implementada pel mètode `identify_language` que, com podem veure amb el *help*, mostra l'estructura següent:

- Input: String que representa o bé un *path* o bé una frase.
- Precondició: En cas que l'input sigui un *path*, aquest ha de ser vàlid.
- Postcondició: Es retorna un vector de prediccions associat a les frases del path o el string. Aquestes prediccions estan en format ISO3 associat al llenguatge en qüestió.

Seguint aquesta estructura, els passos que fa aquesta funció són els següents:

1. Preprocessament del text d'entrada; el mateix que l'aplicat a les dades d'entrenament.
2. Separació per frases.
3. Per cada frase:
   (a) Computem el total de versemblança de cada idioma, amb la suma dels logaritmes de les probabilitats dels trigrames calculats amb els mètodes de Lidstone (Eq. 3.2.1) o *Absolute Discounting* (Eq. 3.2.2).
   (b) Seleccionem l'idioma amb la màxima versemblança com el predit (Eq. 3.1.7).
4. Retornem l'idioma predit per cada frase en forma de vector.

La lògica d'aquest mètode ens permet retornar els valors com qualsevol predictor d'altres llibreries, així sent capaços d'introduir la sortida en, per exemple, mètriques o matrius de confusió. A més a més, funciona especialment bé per fitxers amb de grans dimensions, ja que les processa seqüencialment i retorna un vector de prediccions de la mateixa mida i ordre.

Pel que fa a altres mètodes, n'hi ha d'auxiliars per computar les estimacions de probabilitat, que són privades per l'usuari. No obstant, si que s'ofereix l'opció d'obtenir el logaritme de les probabilitats de què cada idioma hagi generat cert text, amb el mètode `predict_probs`.

5 Anàlisi dels Resultats

En aquest punt, només queda examinar el rendiment del model de classificació, així com possibles biaixos o condicions que l'aboquen a l'error. A més, podrem comparar quin mètode de *smoothing* funciona millor, mitjançant el còrpora reservat pel testeig del model.

Precisió del model

En primera instància, observem que les puntuacions de precisió del model són bastant bones per ambdós models, concretament la mateixa: 0.99876. En definitiva, el mètode de *smoothing* no sembla decisiu en la predicció de les frases de les dades de test. D'altra banda, podem visualitzar les matrius de confusió resultants i concretar quins casos són més crítics pel model.

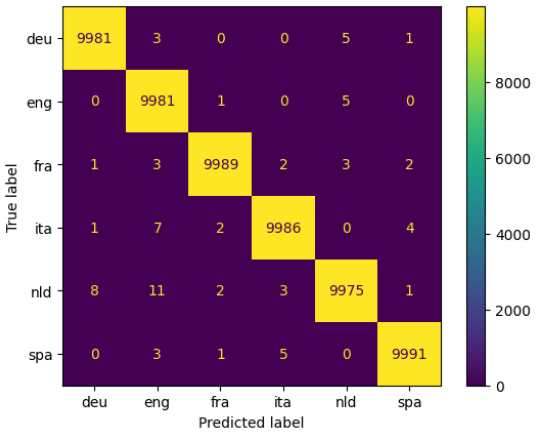

Figure 1: Matriu de confusió amb Lidstone

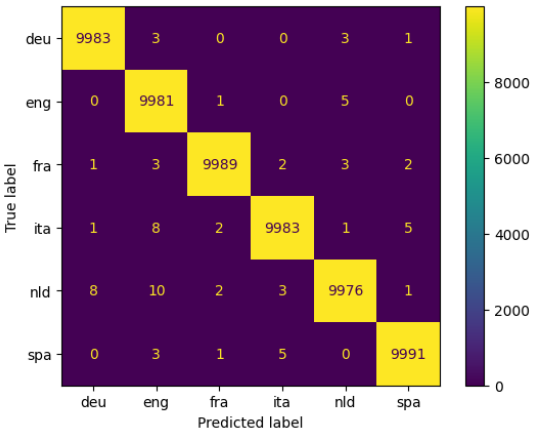

Figure 2: Matriu de confusió amb Absolute Discounting

A partir dels gràfics anteriors, podem identificar un biaix mínim cap a l'anglès, concretament quan l'idioma correcte és italià o neerlandès. A més, l'alemany es confon en alguns casos amb el neerlandès, donades les seves similituds. Pel que fa al francès i l'espanyol, són les classes més encertades pels models, cosa que indica que els trigrames d'aquests idiomes són més característics. No obstant, és interessant veure com reacciona el model quan el forcem amb frases atípiques.

Anàlisi d'errors i biaixos

El model aplicat per a identificar el llenguatge té un entrenament que es basa únicament a mesurar les probabilitats de cada trigrama per cada idioma. A priori, donat que aquestes probabilitats s'aprenen de forma independent, podem assumir que el model no té biaixos majors. Lluny d'això, però, realitzarem una sèrie d'experiments per a comprovar-ho. Tots aquests experiments estan adjuntats amb els fitxers.

En primer lloc, mirarem la freqüència relativa en la qual s'ha predit cada idioma en el test general amb smoothing "Lidstone", segons la fòrmula:

$$f_l = \frac{\sum_{l \in L} p_l}{\sum_{l \in L} p}$$

Els resultats però, no indiquen cap biaix apriori:

Idioma	Freqüència relativa
deu	0.16669
eng	0.16684
fra	0.16663
ita	0.16659
nld	0.16651
spa	0.16671

Table 1: Taula de freqüències relatives.

Un altre apropament serà introduir un test compost per idiomes no presents al *train*, però amb caràcters similars, per no dependre únicament del smoothing. Els idiomes en qüestió són: portuguès, grec, finlandès, i danès. Els resultats han sigut els següents:

Idioma	Freqüència relativa
deu	0.20210
eng	0.0088
fra	0.0065
ita	0.0113
nld	0.5369
spa	0.23242

Table 2: Taula de freqüències relatives amb idiomes no vists.

A primera vista podem veure una gran diferència en les freqüències, però si ens fixem quin idioma s'ha predit per cada un dels idiomes nous es pot descartar la hipòtesi de la presència d'un biaix en l'entrenament. El portuguès ha estat en pràcticament tots els casos confós pel castellà. El danès, el grec i el finlandès majoritàriament pel neerlandès. Un cop vistos els resultats, però, podem concloure que hi ha molta similitud entre els idiomes desconeguts i els predits, on el model que sols estima la versemblança ha escollit en part bastant bé, fent la feina que se li demanava.

Pel que fa a l'anàlisi d'errors, hem pogut trobar coses bastant valuoses. Mitjançant la selecció de les frases mal classificades pel nostre model d'identificació d'idioma, hem pogut caracteritzar les frases que són protagonistes en les errades del nostre model en dos tipus principals.

1. Frases en un idioma que contenen paraules d'un altre idioma, on aquest segon és majoritari i, per tant, el predit. Per exemple:

   - "de volgende frazioni maken deel uit van de gemeente: anzio colonia, marechiaro, cincinnato, falasche, villa claudia, lavinio stazione, lavinio mare, padiglione, lido dei pini." Veiem que la majoria de les paraules són en italià, però la frase forma part del corpus neerlandès.
   - "amy winehouse, su vida en imágenes fotos - shaun curry (afp) - – muere amy winehouse.". On tenim noms en anglès, l'idioma predit, envers el castellà.

2. Frases curtes i, en conseqüència, poc precises probabilísticament. Per exemple:

   - """dal sud"", dice.". Paraula predita com a castellà però realment en italià. És pràcticament interpretable en els dos idiomes.
   - "o sky, o mediaset premium.". Paraula predita com a anglès pels anglicismes, però realment en italià. És tan poc informada que quasi no permet diferenciar.

Ja caracteritzats els dos tipus d'errada, podem extreure la hipòtesi que el primer tipus d'errors és quasi impossible de solucionar amb el nostre tipus de model, que només funciona amb probabilitats que una paraula formi part d'un llenguatge. Encara més, podem intuir que la mida del corpus pot no resoldre aquestes errades. Una mena de "model-bound" errors. En canvi, el segon tipus sembla que es pot anar afinant a mesura que augmentem la mida del corpus, ja que la probabilitat d'aparició dels trigrames s'aproparà cada vegada més a la del llenguatge.

6 Conclusions

En conclusió, l'informe presentat ha proporcionat una justificació detallada de la solució proposada per abordar el problema d'identificació d'idiomes.

En primer lloc, hem discutit el preprocessament de les dades. Aquest ha sigut format per l'eliminació de dígits, la concatenació de frases, la conversió a minúscules i la unificació d'espais. S'ha utilitzat Python juntament amb expressions regulars per a acomplir aquesta tasca de forma eficient i ràpida, ja que era la columna vertebral de la implementació.

A continuació, hem explicat detalladament el funcionament del classificador Naive de Bayes per a la detecció d'idiomes, incloent-hi el plantejament del model de predicció i el càlcul de probabilitats. S'ha destacat l'ús de tècniques de *smoothing* com Lidstone i *Absolute Discounting* per a realitzar l'estimació de les probabilitats.

Finalment, s'ha presentat el procés d'entrenament i de predicció del classificador, així com la tria de paràmetres i l'anàlisi dels resultats obtinguts. Amb aquests resultats, s'ha examinat la precisió del model on s'han mostrat i discutit les matrius de confusió i s'ha fet una anàlisi d'errors i biaixos. S'ha conclòs que el model mostra una precisió elevada, tot i que s'han identificat algunes limitacions i fonts d'errors, com ara la presència de paraules d'un idioma en un text d'un altre idioma, la llargada de les frases en qüestió, o la similitud entre idiomes. I és que el classificador en qüestió assumeix independència entre trigrames, de manera que no pot abordar aquestes situacions favorablement.

En resum, el projecte exposa una implementació de l'identificador d'idiomes eficaç i robusta, tot explicant les possibilitats i les limitacions de l'aproximació proposada.

References

[1] Ney, H., U. Essen, and R. Kneser. 1994. On structuring probabilistic dependencies in stochastic language modelling. Computer Speech and Language, 8:1–38.

[2] Jurafsky, D., & Martin, J. H. (2023). Naive Bayes, Text Classification, and Sentiment. In *Speech and Language Processing*, retrieved from https://web.stanford.edu/~jurafsky/slp3/