DANKMEMES

Fabrizio Meniconi

Relazione progetto per gli esami di Basi di Dati su Larga Scala e Data Mining

e Intelligent Systems

DANKMEMES

EVALITA è una campagna che si occupa di Natural Language Processing (PNL) e degli strumenti vocali per la lingua italiana.

Ogni anno organizza un concorso dove mette a disposizione una serie di dataset con uno o più task da raggiungere su di essi attraverso la loro sottomissione a modelli di Data Mining, i risultati ottenuti vengono poi valutati dalla commissione attraverso un test set di dati non classificato e il modello migliore vince la gara relativa alla challange a cui fa riferimento.

Il progetto descritto in questa relazione è DANKMEMES task1 Meme Recognition preso da EVALITA2020, la cui consegna è descritta nella seguente pagina web: https://dankmemes2020.fileli.unipi.it/.

Consiste nell'istruire un modello per la classificazione di immagini e dati per il riconoscimento di meme politici. L'argomento è stato scelto dal sottoscritto per la centralità e l'importanza nello scenario sociale e politico odierno.

Tecnologie e DataSet

Il dataset fornito consiste in due cartelle: dankmemes_task_1 contenente i dati per la fase di traning e 1_test i dati per il test, ognuna di esse è strutturata nel seguente modo per il train e il test set rispettivamente:

- **images_task1_train/1_test_igm**

  Una cartella contenente immagini in formato .jpg a colori di grandezza variabile.

Conta 1600 immagini per il train set e 400 per il dataset

- **dankmemes_task1_train.csv/meme_test_final.csv**

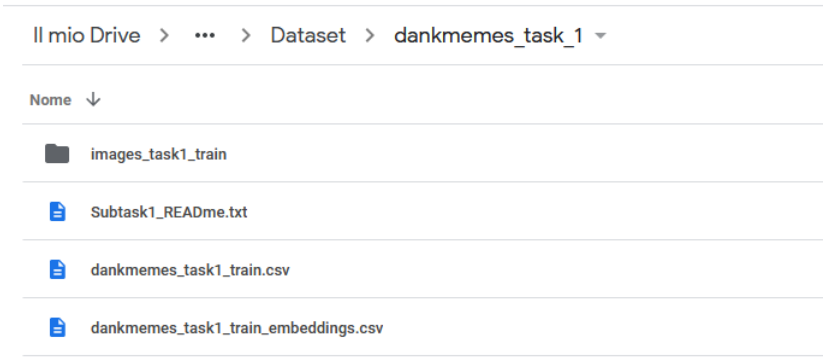
  Un file .csv contenente informazioni relative ad ogni immagine quali: **file** dove viene specificato il nome dell'immagine a cui si fa riferimento, **engagement** che rappresenta il numero dei commenti ricevuti dall'immagine sui social, **date** che espone la data di pubblicazione dell'immagine, **manipulation** che ha valore 1 se l'immagine è stata modificata 0 altrimenti, **visual** che elenca il nome dei politici presenti nell'immagine e **text** che riporta il testo dell'immagine se ne contiene uno.

- **dankmemes_task1_train_embedding.csv/1_embedding.csv**

  Contiene l'embedding delle immagini dei dataset, si tratta i dati di uscita delle immagini sottoposti ad una rete convolutiva chiamata ResNet e consiste in un vettore di 2024 elementi per ogni immagine contenente valori compresi tra 0 e 2.

- **Subtask1_READme.txt**

  Un file .txt che descrive la consegna del progetto e il significato dei file contenuti nel dataset



L'intero progetto è stato realizzato in python nell'editor Google Colab.

I modelli creati sono tutte reti neurali, per i modelli che prendono in analisi le immagini sono state realizzate delle reti convolutive, per quelli che prendono in input sequenze di testo reti per NLP (Natural Language Processing) concatenate con le altre features del dataset.

Ogni modello è stato creato utilizzando la libreria keras di tensorflow.

Data preprocessing

Il data preprocessing è quell'insieme di operazioni, modifiche e accorgimenti a cui vengono sottoposti i dati per far sì che possano essere processate al meglio da un modello.

Nello specifico la prima operazione fatta è stata eliminare le colonne relative alla data di pubblicazione e il nome delle immagini in quanto non significative per il fine del progetto.

Successivamente ogni record della colonna **visual** stata trasformata in un un vettore di 65 elementi ognuno di questi rappresentante di un nome di un politico comparso in almeno un record del train e del test set, l'elemento relativo al politico in questione ha valore 1 se compare nell'immagine a cui riferimento, 0 altrimenti.

Per l'**engagement** i valori sono stati ricalibrati tra 0 e 1 dove 1 rappresenta il valore massimo trovato nel dataset.

Tutte le **immagini** sono state ricalibrate nella dimensione 200 x 200, e successivamente trasformate in matrici (200,200,3) con ogni valore contenuto tra 0 e 1 dove 1 rappresenta il 255 valore della scala RGB.

Infine con il **testo** è stato estrapolato il vocabolario interno del dataset ottenuto con tutte le 7577 parole trovate all'interno di esso attraverso la libreria word_tokenize ed ogni record è stato

trasformato in un vettore di lunghezza pari alla massima lunghezza trovata dove ogni elemento è un numero intero rappresentante la parola all'interno del dataset, 0 se non c'è nessuna parola.

Questi vettori sono stati ridotti ad una lunghezza di 50 in quanto solo il 2,5% dei record supera tale soglia, questa operazione garantisce, al costo di perdere pochi dati significativi, di eliminare moltissimi 0 non significativi in ogni vettore garantendo maggiore velocità nel training e il test dei modelli che ne fanno uso.

E' stata anche creata una matrice di Embedding per i testi che sarà poi usata nelle reti ricorrenti, consiste in una serie di pesi associati alle parole del vocabolario italiano, parole semanticamente più rilevanti avranno un peso maggiore all'interno del modello a discapito di quelle meno significative come verbi, congiunzioni, articoli e proposizioni. questa matrice è stata creata mettendo in relazione un insieme di vettori di 300 valori per ogni parola scaricato dalla pagina https://fasttext.cc/docs/en/crawl-vectors.html e la dimensione del vocabolario interno.

Va segnalato che non è stato possibile scaricare valori di Embedding per le immagini di test nella loro interezza in quanto non si è riuscito ad estrapolare le informazioni relative al primo elemento, quindi per ogni colonna di test è stata creata una copia senza la prima riga riducendo la grandezza a 399 per poterle utilizzare nei modelli che fanno uso di questa rappresentazione delle immagini.

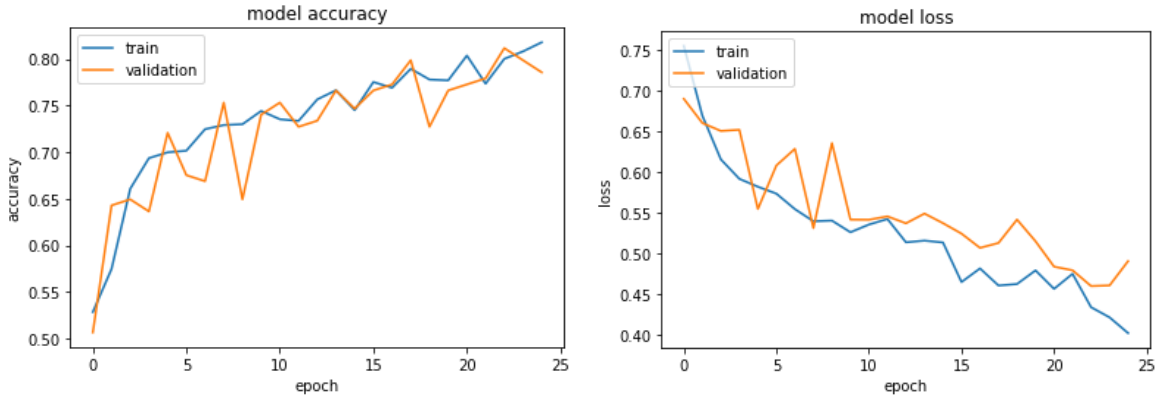
Costruzione dei modelli

Sono state costruite tre topologie di reti: la prima prende in esame solo le immagini, la seconda solamente i valori relativi a testo, manipulation, visual ed engagement, infine l'ultimo che prende in esame tutti gli elementi del dataset.

Per stimarne la validità ogni modello si è fatto uso della Cross Validation, durante l'addestramento il traning set viene diviso in 5 sottoinsiemi, quattro di essi sono usati per allenare il modello, l'ultimo per verificarne i risultati calcolando l?accuracy e il valore della loss function. L'operazione viene ripetuta finché tutti i sottoinsiemi non hanno fatto da test set.

Una volta completate le 5 fasi di traning viene calcolata la media e la varianza dell'accuracy e la media del loss di tutti gli addestramenti, il modello migliore massimizza l'accuracy e minimizza la loss function.

Durante l'addestramento vengono anche calcolati i valori validation accuracy e il validation loss per ogni epoca che, messi in relazione con i valori dell'accuracy e della loss corrispondenti, permettono di ricavare, per ogni modello, i seguenti grafici:



I valori del traning set e della validation devono rimanere coerenti, quando la validation accuracy smette di crescere insieme aall'accuracy e quando la validation loss inizia ad aumentare mentre la loss continua a crescere significa che il nostro modello sta andando in overfitting. Per

prevenire ciò è importante fermare il modello alla giusta epoca prima che questo fenomeno si manifesti o in alternativa cercare di costruire un modello meno complesso.

- **Modelli delle immagini**

  Sono tre modelli diversi i primi due sono due reti convolutive prendono in input i valori di ogni immagine e le processano attraverso livelli Convolutivi e di Pooling, differiscono per la quantità di filtri a cui vengono sottoposte e il numero di livelli convolutivi, dopo il livello di flatten che srotola le matrici risultanti entrambe passano per un livello fully connected prima di arrivare all'ultimo strato dense di cardinalità 1 che restituisce il risultato.

  Il terzo modello è una semplice rete neurale che prende come input i vettori di Embedding delle immagini.

  I risultati della Cross Validation dei modelli viene riportato di seguito:

Primo modello:

```
------------------------------------------------------------------------
Score per fold
------------------------------------------------------------------------
> Fold 1 - Loss: 0.5427683591842651 - Accuracy: 0.753125011920929
------------------------------------------------------------------------
> Fold 2 - Loss: 0.48359426856040955 - Accuracy: 0.7875000238418579
------------------------------------------------------------------------
> Fold 3 - Loss: 0.5798927545547485 - Accuracy: 0.7250000238418579
------------------------------------------------------------------------
> Fold 4 - Loss: 0.512194037437439 - Accuracy: 0.737500011920929
------------------------------------------------------------------------
> Fold 5 - Loss: 0.563015341758728 - Accuracy: 0.75
------------------------------------------------------------------------
Average scores for all folds:
> Accuracy: 0.7506250143051147 (+- 0.02095381961897382)
> Loss: 0.536292952299118
------------------------------------------------------------------------
```

Secondo modello:

```
------------------------------------------------------------------------
Score per fold
------------------------------------------------------------------------
> Fold 1 - Loss: 0.5168489217758179 - Accuracy: 0.737500011920929
------------------------------------------------------------------------
> Fold 2 - Loss: 0.5561203360557556 - Accuracy: 0.7281249761581421
------------------------------------------------------------------------
> Fold 3 - Loss: 0.5766714811325073 - Accuracy: 0.7281249761581421
------------------------------------------------------------------------
> Fold 4 - Loss: 0.5161513090133667 - Accuracy: 0.7718750238418579
------------------------------------------------------------------------
> Fold 5 - Loss: 0.5450651049613953 - Accuracy: 0.762499988079071
------------------------------------------------------------------------
Average scores for all folds:
> Accuracy: 0.7456249952316284 (+- 0.01817881225263643)
> Loss: 0.5421714305877685
------------------------------------------------------------------------
```

Terzo Modello:

```
------------------------------------------------------------------------
Score per fold
------------------------------------------------------------------------
> Fold 1 - Loss: 0.5168780088424683 - Accuracy: 0.7281249761581421
------------------------------------------------------------------------
> Fold 2 - Loss: 0.48142433166503906 - Accuracy: 0.762499988079071
------------------------------------------------------------------------
> Fold 3 - Loss: 0.5030052661895752 - Accuracy: 0.784375011920929
------------------------------------------------------------------------
> Fold 4 - Loss: 0.507491409778595 - Accuracy: 0.778124988079071
------------------------------------------------------------------------
> Fold 5 - Loss: 0.5029298067092896 - Accuracy: 0.7593749761581421
------------------------------------------------------------------------
Average scores for all folds:
> Accuracy: 0.762499988079071 (+- 0.0195656047039278)
> Loss: 0.5023457646369934
------------------------------------------------------------------------
```

I valori della Cross Validation dei tre modelli sono sintetizzati nella seguente tabella:

	Mean Accuracy	Mean Loss	Standard Deviation Mean
Image Model 1	0.7506250143051147	0.536292952299118	0.02095381961897382
Image Model 2	0.7456249952316284	0.5421714305877685	**0.01817881225263643**
Image Model 3 (Embedding)	**0.762499988079071**	0.**5023457646369934**	0.0195656047039278

Si può evincere che il modello con l'embedding delle immagini è il più performante anche se presenta una deviazione standard leggermente superiore al secondo modello

- **Modelli Features**

  Sono due reti neurali per NLP che prendono in input le sequenze di testo, il primo livello di Embedding viene inizializzato con i pesi del modello pre-allenato inseriti nell'embedding matrix creata nella fase il preprocessing. il risultato passa attraverso uno o più livelli convolutivi e di pooling 1D. Dopo il livello flatten il risultato viene concatenato ai valori visual, engagement e manipulation e si hanno gli ultimi livelli dense per arrivare all'ultimo strato di cardinalità 1.

  I risultati della Cross Validation dei modelli viene riportato di seguito:

Primo modello:

```
------------------------------------------------------------------------
Score per fold
------------------------------------------------------------------------
> Fold 1 - Loss: 0.551555335521698 - Accuracy: 0.7281249761581421
------------------------------------------------------------------------
> Fold 2 - Loss: 0.5298953652381897 - Accuracy: 0.731249988079071
------------------------------------------------------------------------
> Fold 3 - Loss: 0.5497488975524902 - Accuracy: 0.7437499761581421
------------------------------------------------------------------------
> Fold 4 - Loss: 0.5164955854415894 - Accuracy: 0.7281249761581421
------------------------------------------------------------------------
> Fold 5 - Loss: 0.522071897983551 - Accuracy: 0.765625
------------------------------------------------------------------------
Average scores for all folds:
> Accuracy: 0.7393749833106995 (+- 0.014334188557312198)
> Loss: 0.5339534163475037
------------------------------------------------------------------------
```

Secondo modello:

```
------------------------------------------------------------------------
Score per fold
------------------------------------------------------------------------
> Fold 1 - Loss: 0.5454359650611877 - Accuracy: 0.731249988079071
------------------------------------------------------------------------
> Fold 2 - Loss: 0.5635732412338257 - Accuracy: 0.7124999761581421
------------------------------------------------------------------------
> Fold 3 - Loss: 0.5866557955741882 - Accuracy: 0.7281249761581421
------------------------------------------------------------------------
> Fold 4 - Loss: 0.5705175399780273 - Accuracy: 0.703125
------------------------------------------------------------------------
> Fold 5 - Loss: 0.5639997720718384 - Accuracy: 0.7124999761581421
------------------------------------------------------------------------
Average scores for all folds:
> Accuracy: 0.7174999833106994 (+- 0.010569705694791941)
> Loss: 0.5660364627838135
------------------------------------------------------------------------
```

I valori della Cross Validation dei due modelli sono sintetizzati nella seguente tabella:

	Mean Accuracy	Mean Loss	Standard Deviation Mean
Features Model 1	**0.7393749833106995**	**0.5339534163475037**	0.014334188557312198
Features Model 2	0.7174999833106994	0.5660364627838135	**0.010569705694791941**

Il modello più performante si è rivelato il primo, quello con un solo layer convolutivo 1D.

- **Modello Immagini più features**

  Due modelli che tentano di concatenare al meglio i migliori modelli emersi dalle tipologie precedenti, differiscono per numero di filtri nei livelli convolutivi 1D e numero di percettroni nei livelli finali fully-connected. Il primo presenta dei livelli dense prima della concatenazione per i dati features e delle immagini.

  I risultati della Cross Validation dei modelli viene riportato di seguito:

Modello Finale 1:

```
------------------------------------------------------------------------
Score per fold
------------------------------------------------------------------------
> Fold 1 - Loss: 0.3928939700126648 - Accuracy: 0.8125
------------------------------------------------------------------------
> Fold 2 - Loss: 0.46842366456985474 - Accuracy: 0.7906249761581421
------------------------------------------------------------------------
> Fold 3 - Loss: 0.388723760843277 - Accuracy: 0.824999988079071
------------------------------------------------------------------------
> Fold 4 - Loss: 0.431366503238678 - Accuracy: 0.8125
------------------------------------------------------------------------
> Fold 5 - Loss: 0.3921840786933899 - Accuracy: 0.828125
------------------------------------------------------------------------
Average scores for all folds:
> Accuracy: 0.8137499928474426 (+- 0.013199201373306506)
> Loss: 0.41471839547157285
------------------------------------------------------------------------
```

Modello Finale 2:

```
------------------------------------------------------------------------
Score per fold
------------------------------------------------------------------------
> Fold 1 - Loss: 0.4625823497772217 - Accuracy: 0.7718750238418579
------------------------------------------------------------------------
> Fold 2 - Loss: 0.45690497756004333 - Accuracy: 0.796875
------------------------------------------------------------------------
> Fold 3 - Loss: 0.46564358472824097 - Accuracy: 0.784375011920929
------------------------------------------------------------------------
> Fold 4 - Loss: 0.47022780776023865 - Accuracy: 0.78125
------------------------------------------------------------------------
> Fold 5 - Loss: 0.42725858092308044 - Accuracy: 0.8218749761581421
------------------------------------------------------------------------
Average scores for all folds:
> Accuracy: 0.7912500023841857 (+- 0.017275328951442206)
> Loss: 0.456523460149765
------------------------------------------------------------------------
```

I valori della Cross Validation dei due modelli sono sintetizzati nella seguente tabella:

	Mean Accuracy	Mean Loss	Standard Deviation Mean
Final Model 1	**0.8137499928474426**	**0.41471839547157285**	**0.013199201373306506**
Final Model 2	0.7912500023841857	0.456523460149765	0.017275328951442206

Il modello più performante si è rivelato il primo, quello con i livelli fully-connected prima della concatenazione.

La tipologia con immagini + features è quella che ha ottenuto in assoluto i risultati migliori.

Risultati

In base ai risultati delle Cross Validation viene scelto il miglior modello per ogni tipologia, ognuno di questi viene riallenato utilizzando l'intero Trainset, successivamente viene valutato utilizzando il test set fornito da evalita, i parametri per la valutazione delle performance che se ne ricavano sono calcolati attraverso la libreria Metrics di Sklearn.

Gli indici in questione sono precision, recall, accuracy e f1 e sono calcolate come segue:

$$\text{Accuracy} = \frac{TP+TN}{TP+TN+FP+FN}$$

$$\text{Precision}\ (p) = \frac{TP}{TP+FP}$$

$$\text{Recall}\ (r) = \frac{TP}{TP+FN}$$

$$\text{F-measure}\ (F) = \frac{2 \cdot r \cdot p}{r+p} = \frac{2 \cdot TP}{2 \cdot TP+FN+FP}$$

Dove TP, TN, FP e FN rappresentano rispettivamente gli insiemi dei true positive, true negative, false positive e false negative ricavate dalla matrice di confusione dopo la fase di testing.

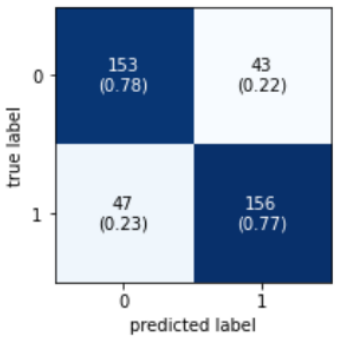
I risultati ottenuti per ogni modello sono:

- **Modello Immagini**

  Dal miglior modello delle immagini sono emersi i seguenti risultati:

```
Accuracy: 0.774436
Precision: 0.783920
Recall: 0.768473
F1 score: 0.776119
```

Confusion matrix:

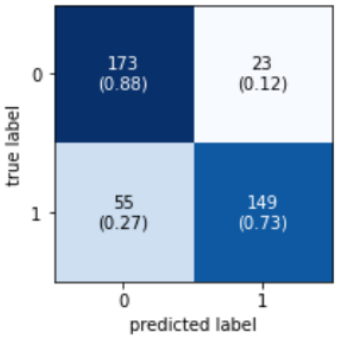


- **Modello Features**

  Dal miglior modello delle features sono emersi i seguenti risultati::

```
Accuracy: 0.805000
Precision: 0.866279
Recall: 0.730392
F1 score: 0.792553
```

Confusion matrix:

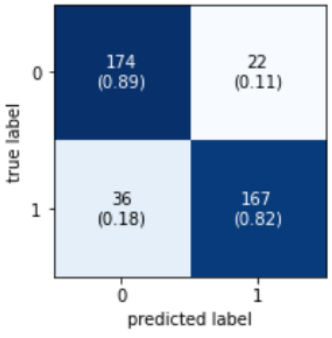


- **Modello Immagini più Features**

  Dal miglior modello delle immagini concatenate alle features sono emersi i seguenti risultati::

```
Accuracy: 0.854637
Precision: 0.883598
Recall: 0.822660
F1 score: 0.852041
```

Confusion matrix:



Tutti i risultati ottenuti vengono riportati nella seguente tabella:

	Accuracy	Precision	Recall	f1 score
Image Model	0.774436	0.783920	0.768473	0.776119
Features Model	0.805000	0.866279	0.730392	0.792553
Final Model	**0.854637**	**0.883598**	**0.822660**	**0.852041**

Conclusioni

Durante la fase di traning è emerso che quelli che presentano un numero inferiore di filtri nei prodotti di convoluzione e con livelli di dense che hanno meno percettroni performano meglio. Questo fenomeno è dovuto alla dimensione contenuta del traning set che può mandare facilmente in overfitting i modelli troppo complessi.

I livelli di dropout hanno contribuito a creare modelli più performanti, ritardando l'overfitting e abbassando notevolmente la varianza dei risultati. Livelli di dropout che eliminano una considerevole quantità di percettroni dal livello sono stati particolarmente utili nelle parti del modello che presentano dati pre allenati come l'embedding delle immagini e l'embedding del testo.

Come da previsione il modello che fa uso di tutti i dati del dataset si è rivelato il migliore, questo è sintomo che i dati forniti erano utili alla classificazione e avevano un basso livello di ridondanza tra loro.

Il modello features ha ottenuto risultati migliori rispetto a quello delle immagini ma un Recall minore anche se le stime della Cross Validation facevano pensare il contrario, probabilmente perché il modello delle immagini aveva una deviazione standard maggiore ed è quindi meno affidabile.

Quello che è emerso nella fase di test è stato confrontato con i risultati ottenuti dai partecipanti al concorso originale pubblicati nella nella pagina web http://ceur-ws.org/Vol-2765/, dove di seguito viengono riportati due esempi:

Esempio 1:

Model	**Pr**	**Re**	**F1**
random baseline	0.525	0.5147	0.5198
Basic Features	**0.8732**	0.6078	0.7168
BF+fastText	0.8253	0.6716	0.7405
BF+GilBERTo	0.7685	0.7647	0.7666
BF+ResNet	0.8341	0.8382	0.8362
BF+fastText+ ResNet (**SNK1**)	0.8515	0.8431	**0.8473**
BF+GilBERTo+ ResNet (**SNK2**)	0.8317	**0.848**	0.8398

Esempio 2:

Team	Run	Recall	Precision	F_1
Unitor	2	0.8522	0.848	0.8501
SNK	1	0.8515	0.8431	0.8473
UPB	2	0.8543	0.8333	0.8437
Unitor	1	0.839	0.8431	0.8411
SNK	2	0.8317	0.848	0.8398
UPB	1	0.861	0.7892	0.8235
DMT	1	0.8249	0.7157	07664
Keila	1	0.8121	0.6569	0.7263
Keila	2	0.7389	0.652	0.6927
baseline	1	0.525	0.5147	0.5198

Possiamo reputare quanto ottenuto dal nostro modello finale un risultato più che soddisfacente in quanto in linea con i migliori risultati dei modelli in gara.