ΠΑΝΕΠΙΣΤΗΜΙΟ ΠΕΙΡΑΙΩΣ
ΠΜΣ ΠΛΗΡΟΦΟΡΙΑΚΑ ΣΥΣΤΗΜΑΤΑ ΚΑΙ ΥΠΗΡΕΣΙΕΣ
ΜΕΓΑΛΑ ΔΕΔΟΜΕΝΑ ΚΑΙ ΑΝΑΛΥΤΙΚΗ

Ανάλυση συναισθήματος χρησιμοποιώντας state-of-the-art τεχνικές

Ευαγγελιδάκης Λέανδρος

Table of Contents

1. Ιδέα έργου

Πλέον, υπολογίζεται ότι το 80% των δεδομένων είναι μη-δομημένα (unstructured, π.χ. κείμενα). Καθημερινά δημιουργούνται χιλιάδες νέα tweets, σχόλια σε διαδικτυακές πλατφόρμες, κριτικές προϊόντων, emails κ.λπ. Οι εταιρίες έχουν ανάγκη να επεξεργάζονται τέτοιου είδους πληροφορίες για να εξάγουν χρήσιμες γνώσεις σχετικά με τα προϊόντα τους.
Στην παρούσα εργασία εξετάζουμε τεχνικές ανάλυσης συναισθήματος σε αρχεία κειμένου. Με τον όρο «ανάλυση συναισθήματος» (sentiment analysis) αναφερόμαστε σε διάφορες μεθόδους, οι οποίες στοχεύουν στο να αναγνωρίσουμε το συναίσθημα, ή την άποψη, που εμπεριέχεται σε ένα κείμενο. Αναλύουμε κριτικές ταινιών με σκοπό να τις κατηγοριοποιήσουμε ως θετικές ή ως αρνητικές, δηλαδή έχουμε ένα πρόβλημα τύπου binary text classification. Ο σκοπός είναι να μελετήσουμε κάποιους αλγορίθμους που χαρακτηρίζονται αυτή τη στιγμή ως state-of-the-art ως προς την ακρίβεια της κατηγοριοποίησης κειμένου φυσικής γλώσσας. Παρόλο που πλέον οι αλγόριθμοι είναι ιδιαίτερα περίπλοκοι και απαρτίζονται, συνήθως, από διάφορες σύνθετες αρχιτεκτονικές νευρωνικών δικτύων, με τα σημερινά τεχνολογικά εργαλεία που είναι διαθέσιμα, η χρήση τους είναι σημαντικά ευκολότερη απ'ότι ήταν κάποια χρόνια πριν.

2. Μέθοδοι / Τεχνικές

Παρακάτω περιγράφουμε συνοπτικά κάποια βασικά χαρακτηριστικά των αλγορίθμων που χρησιμοποιήσαμε.

2.1 NBSVM (Naïve Bayes - Support Vector Machine)

Οι αλγόριθμοι **N**aïve **B**ayes (**NB**) και **S**upport **V**ector **M**achines(**SVM**), αποτελούν δύο από τους δημοφιλέστερους αλγορίθμους για κατηγοριοποίηση κειμένου. Το 2012, οι Wang και Manning από το πανεπιστήμιο Stanford, δημοσίευσαν ένα άρθρο **[1],** όπου περιέγραψαν μια νέα μέθοδο για ανάλυση συναισθήματος, στην οποία συγχωνεύονται οι αλγόριθμοι NB και SVM. Όπως έδειξαν, χρησιμοποιώντας το Multinomial Naïve Bayes (MNM) πετυχαίνουμε καλύτερα αποτελέσματα σε μικρού μήκους κείμενα, ενώ με Support Vector Machines (SVM) σε μεγάλου μήκους. Συνδυάζοντας τα δύο παραπάνω πετυχαίνουμε καλύτερα αποτελέσματα και για τις δύο περιπτώσεις. Η γενική ιδέα που ακολουθεί το μοντέλο **NBSVM** είναι ότι χρησιμοποιούνται Support Vector Machines, στα ποσοστά (λογαριθμισμένα) που προκύπτουν από τον αλγόριθμο Naïve Bayes. Επιπλέον, έδειξαν ότι χρησιμοποιώντας bigrams (2-grams), αυξάνεται περεταίρω η ακρίβεια του αλγορίθμου. Ο NBSVM θεωρείται μια ισχυρή baseline για την εκτίμηση της αποτελεσματικότητας των κλασσικών αλγορίθμων της μηχανικής μάθησης.

2.2 FASTTEXT

Μολονότι τα νευρωνικά δίκτυα παρουσιάζουν τα καλύτερα αποτελέσματα στα προβλήματα Ανάλυσης Φυσικής Γλώσσας (NLP), απαιτούν αρκετό χρόνο εκπαίδευσης και στον τομέα των big-data αυτό είναι ένα σημαντικό πρόβλημα. Με κίνητρο το ζήτημα αυτό, το 2016, η Facebook παρουσίασε τη μέθοδο *fastText* **[2]** η οποία, όπως προϊδεάζει και το όνομά της, απαιτεί αρκετά λιγότερο χρόνο εκπαίδευσης. Τα embeddings προκύπτουν παίρνοντας το μέσο όρο των bag-of-ngrams αντί για bag-of-words. Με αυτό τον τρόπο διατηρείται η αποδοτικότητα του αλγορίθμου χωρίς να επηρεάζεται η ακρίβεια. Επιπλέον, χρησιμοποιούνται τεχνικές (hashing-trick) για την αποδοτική αποθήκευση των ngrams στη μνήμη ώστε να πετυχαίνονται μικροί χρόνοι προσπέλασης των χαρακτηριστικών κατά τη διαδικασία της εκπαίδευσης. Στη συνέχεια εφαρμόζεται λογιστική παλινδρόμηση χρησιμοποιώντας τα embeddings ως χαρακτηριστικά (features).

2.3 BERT (Bidirectional Encoder Representations from Transformers)

Πρόκειται για μια νέα μέθοδο (2019) για την προ-εκπαίδευση γλωσσικών αναπαραστάσεων (language representations) από την Google **[3]**. Το μοντέλο BERT, εξετάζει τα δεδομένα από αριστερά προς τα δεξιά αλλά και το αντίστροφο (Bidirectional). Έτσι, αποτυπώνει πολύ καλύτερα το περιεχόμενο/συμφραζόμενα (context) του κειμένου που εξετάζει, στα embeddings που παράγει. Το μοντέλο προεκπαιδεύεται αρχικά, μέσω Masked Language Models, μοντέλα που εξάγουν τυχαία ένα στοιχείο από ένα σύνολο (π.χ. μια λέξη από μια πρόταση) και προσπαθούν να κάνουν πρόβλεψη αυτής με βάση μόνο τα συμφραζόμενα. Οι διαδικασίες αυτές χρησιμεύουν στο να εκπαιδευτεί ένας bidirectional Transformer. Επιπλέον, χρησιμοποιούνται μέθοδοι για πρόβλεψη επόμενης πρότασης ως επιπρόσθετη προ-εκπαίδευση.
Η προ-εκπαίδευση αυτή έγινε σε ολόκληρη τη Wikipedia (+ BookCorpus). Χρησιμοποιώντας τα προ-εκπαιδευμένα μοντέλα αυτά, αρκεί να κάνουμε fine-tuning στα δεδομένα του εκάστοτε προβλήματος. Στην υλοποίησή μας το μοντέλο BERT που χρησιμοποιούμε περιλαμβάνει :
12-layer, 768-hidden, 12-heads, 100M parameters

2.3.1 DISTILBERT

Εκτός από το κλασσικό μοντέλο BERT χρησιμοποιούμε μια παραλλαγή του, το μοντέλο DistilBERT **[4]** . Πρόκειται για μια light έκδοση του BERT, η οποία έχει μειωμένο μέγεθος κατά 40%, ενώ διατηρεί το 99% της απόδοσης του αρχικού μοντέλου και τρέχει 60% ταχύτερα.

4. Πειραματική διαδικασία

4.1 Δεδομένα

Τα δεδομένα προέρχονται από την βάση δεδομένων του Stanford και βρίσκονται στην παρακάτω τοποθεσία : http://ai.stanford.edu/~amaas/data/sentiment/

Στο αρχείο περιέχονται 25.000 κριτικές για εκπαίδευση (training) και 25.000 κριτικές για έλεγχο (testing). Τα δεδομένα δεν είναι καθαρισμένα και είναι όλα στην Αγγλική γλώσσα.

4.2 Εργαλεία-τεχνολογίες

Η υλοποίηση της εργασίας έγινε με τη γλώσσα προγραμματισμού Python στο περιβάλλον Google Colaboratory. Οι σημαντικότερες βιβλιοθήκες που χρησιμοποιήθηκαν είναι οι Numpy και Pandas για την επεξεργασία των δεδομένων και tensorflow, keras, ktrain για την εφαρμογή των μοντέλων μηχανικής μάθησης.

4.3 Προεπεξεργασία

Για την προεπεξεργασία των δεδομένων χρησιμοποιούμε τις παρακάτω διαδικασίες:

- Αφαίρεση των HTML ετικετών (tags)
- Αφαίρεση κενών
- Αφαίρεση ειδικών χαρακτήρων (special characters)
- Stemming (Αποκοπή καταλήξεων), για ορισμένους αλγορίθμους

4.4 Εφαρμογή αλγορίθμων

Για κάθε αλγόριθμο που θα χρησιμοποιήσουμε, ακολουθούμε την εξής διαδικασία:

- Μετατροπή των δεδομένων στην κατάλληλη μορφή που απαιτεί κάθε αλγόριθμος
- Επιλογή ρυθμού μάθησης (learning rate tuning)
- Εκπαίδευση αλγορίθμου στο train set και επαλήθευση στο test set

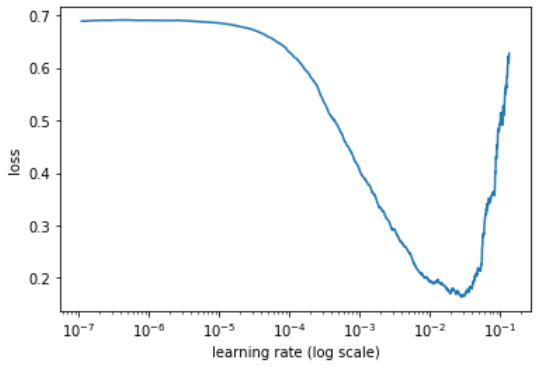
4.4.1 NBSVM

Για την εφαρμογή του NBSVM προσθέτουμε επιπλέον τα 2-grams κάθε κειμένου. Η υλοποίηση του αλγορίθμου είναι μια προσέγγιση με νευρωνικά δίκτυα.

Εξέταση ρυθμού μάθησης

```
simulating training for different learning rates... this may take a few moments...
Train on 25000 samples
Epoch 1/1024
25000/25000 [==============================] - 10s 404us/sample - loss: 0.6664 - accuracy: 0.6862
Epoch 2/1024
20416/25000 [=======================>......] - ETA: 1s - loss: 0.3316 - accuracy: 0.9232

done.
```



Για τον καθορισμό του learning rate (lr) χρησιμοποιούμε Stochastic Gradient Decent with Restarts (SGDR) **[5]** . Παρακάτω φαίνονται τα αποτελέσματα από την εκπαίδευση με lr>0.001 και για 5 epochs.

```
Train on 25000 samples, validate on 25000 samples
Epoch 1/5
25000/25000 [==============================] - 12s 463us/sample - loss: 0.2134 - accuracy: 0.9333 - val_loss: 0.2402 - val_accuracy: 0.9218
Epoch 2/5
25000/25000 [==============================] - 12s 463us/sample - loss: 0.0760 - accuracy: 0.9832 - val_loss: 0.2372 - val_accuracy: 0.9154
Epoch 3/5
25000/25000 [==============================] - 11s 455us/sample - loss: 0.0435 - accuracy: 0.9941 - val_loss: 0.2320 - val_accuracy: 0.9189
Epoch 4/5
25000/25000 [==============================] - 12s 455us/sample - loss: 0.0281 - accuracy: 0.9972 - val_loss: 0.2299 - val_accuracy: 0.9163
Epoch 5/5
25000/25000 [==============================] - 12s 487us/sample - loss: 0.0193 - accuracy: 0.9987 - val_loss: 0.2293 - val_accuracy: 0.9157
<tensorflow.python.keras.callbacks.History at 0x7fdd625b29b0>
```

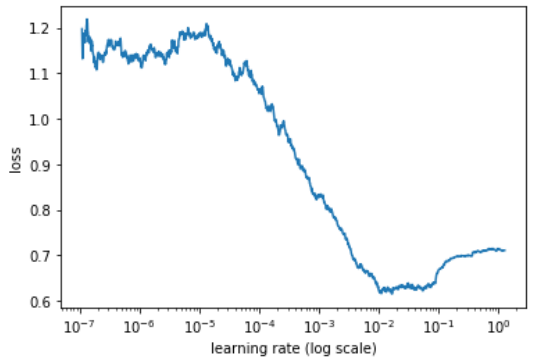
Το μοντέλο πετυχαίνει ακρίβεια 91.57% και όπως φαίνεται από τους χρόνους εκπαίδευσης λειτουργεί ιδιαίτερα γρήγορα. (≈ 11-12s / epoch)

4.4.2 FASTTEXT

Εξέταση ρυθμού μάθησης

```
simulating training for different learning rates... this may take a few moments...
Train on 25000 samples
Epoch 1/1024
25000/25000 [==============================] - 431s 17ms/sample - loss: 1.1210 - accuracy: 0.5011
Epoch 2/1024
25000/25000 [==============================] - 431s 17ms/sample - loss: 0.7065 - accuracy: 0.5934
Epoch 3/1024
 2720/25000 [==>...........................] - ETA: 6:23 - loss: 22.6590 - accuracy: 0.5070

done.
```



Εκπαίδευση με lr<=0.01 για 3 epochs, χρησιμοποιώντας μια παραλλαγή του triangular learning rate policy **[6].**

```
begin training using triangular learning rate policy with max lr of 0.01...
Train on 25000 samples, validate on 25000 samples
Epoch 1/3
25000/25000 [==============================] - 419s 17ms/sample - loss: 0.5938 - accuracy: 0.6886 - val_loss: 0.3980 - val_accuracy: 0.8383
Epoch 2/3
25000/25000 [==============================] - 413s 17ms/sample - loss: 0.3941 - accuracy: 0.8262 - val_loss: 0.3341 - val_accuracy: 0.8569
Epoch 3/3
25000/25000 [==============================] - 410s 16ms/sample - loss: 0.2986 - accuracy: 0.8756 - val_loss: 0.3245 - val_accuracy: 0.8678
<tensorflow.python.keras.callbacks.History at 0x7f4ee2f14c88>
```

Επανάληψη για άλλες 3

```
begin training using triangular learning rate policy with max lr of 0.001...
Train on 25000 samples, validate on 25000 samples
Epoch 1/3
25000/25000 [==============================] - 410s 16ms/sample - loss: 0.2253 - accuracy: 0.9090 - val_loss: 0.3195 - val_accuracy: 0.8698
Epoch 2/3
25000/25000 [==============================] - 404s 16ms/sample - loss: 0.2071 - accuracy: 0.9152 - val_loss: 0.3196 - val_accuracy: 0.8723
Epoch 3/3
25000/25000 [==============================] - 400s 16ms/sample - loss: 0.1942 - accuracy: 0.9241 - val_loss: 0.3210 - val_accuracy: 0.8729
<tensorflow.python.keras.callbacks.History at 0x7f4eda0f8630>
```

Τελική ακρίβεια : 87.29%, συνολικός χρόνος 2456s (≈ 410s / epoch)

4.4.3 BERT

Το μοντέλο BERT της Google έρχεται προεκπαιδευμένο και εμείς εκπαιδεύουμε ένα τελικό layer. Οι χρόνοι εκπαίδευσης είναι ιδιαίτερα μεγάλοι και έτσι είναι δύσκολη η εκπαίδευση για πολλά epochs. Για τον ρυθμό μάθησης χρησιμοποιούμε την τιμή $2*10^{-5}$, που όπως αναφέρει η Google είναι γενικά μια καλή επιλογή. Εκπαιδεύουμε το μοντέλο για 1 epoch, χρησιμοποιώντας 1cycle policy για την προσαρμογή του learning rate.

Παρακάτω φαίνονται τα αποτελέσματα.

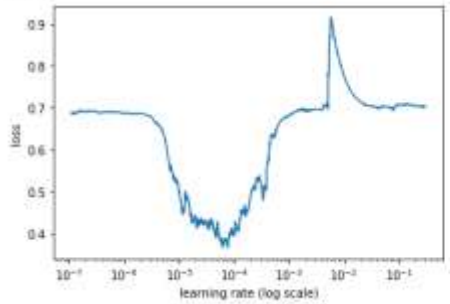
```
begin training using onecycle policy with max lr of 2e-05...
Train on 25000 samples, validate on 25000 samples
25000/25000 [==============================] - 8466s 338ms/sample - loss: 0.1282 - accuracy: 0.9545 - val_loss: 0.1590 - val_accuracy: 0.9424
<tensorflow.python.keras.callbacks.History at 0x7f4ef61b58d0>
```

Όπως φαίνεται, με μόλις 1 εποχή εκπαίδευσης, πετυχαίνουμε ακρίβεια 94.24%. Επίσης φαίνεται και ο χρόνος εκπαίδευσης που ήταν 8466s .

4.4.4 DistilBERT

Καθορισμός ρυθμού μάθησης

```
simulating training for different learning rates... this may take a few moments...
Train for 4166 steps
Epoch 1/1024
1506/4166 [==========>.....................] - ETA: 30:40 - loss: 1.0093 - accuracy: 0.6034
done.
```



Χρησιμοποιούμε την ίδια διαδικασία όπως και στο BERT για learning rate <= $5*10^{-5}$

```
begin training using onecycle policy with max lr of 5e-05...
Train for 4167 steps, validate for 4167 steps
4167/4167 [==============================] - 3828s 919ms/step - loss: 0.2917 - accuracy: 0.8764 - val_loss: 0.1679 - val_accuracy: 0.9255
<tensorflow.python.keras.callbacks.History at 0x7f4ee0c2b908>
```

Όπως φαίνεται, το μοντέλο έχει ακρίβεια 92.55% και η διαδικασία έγινε σε 3828s.

5. Αποτελέσματα

Συνολικά τα αποτελέσματα του κάθε αλγορίθμου φαίνονται στον παρακάτω Πίνακα 1.

Αλγόριθμος	Accuracy	Loss	Χρόνος Εκπαίδευσης	Epochs	Χρόνος / Epoch
NBSVM (+bi-grams)	0.9157	0.2293	59s	5	12s
FASTTEXT (+bi-grams)	0.8729	0.3245	2456s	6	410s
BERT	0.9424	0.1590	8466s	1	8466s
DISTILBERT	0.9255	0.1879	3828s	1	3828s

Πίνακας 1. Ακρίβεια κατηγοριοποίησης των αλγορίθμων

Όπως φαίνεται, το μοντέλο BERT παράγει τα καλύτερα αποτελέσματα. Στα πλαίσια της εργασίας η εκπαίδευσή του έγινε για μόνο μία epoch. Εκπαιδεύοντάς το για περεταίρω epochs μπορούμε να πετύχουμε ακόμη καλύτερα αποτελέσματα.

Πρακτική εφαρμογή

Εκτός από την εφαρμογή στο σύνολο δεδομένων του IMDB που χρησιμοποιήσαμε, παραθέτουμε κάποια πρακτικά παραδείγματα σε δικό μας κείμενο . Στον κώδικα που συνοδεύεται η εργασία, έχουμε φτιάξει μια συνάρτηση όπου ο χρήστης εισάγει κάποια κείμενα και γίνεται η κατηγοριοποίησή τους ως θετικά ή αρνητικά. Με αυτόν τον τρόπο εξετάσαμε τους αλγορίθμους για διάφορα παραδείγματα, ώστε να πάρουμε μια πιο συνολική εικόνα για τον τρόπο που δουλεύουν. Μελλοντικά αυτό θα μπορούσε να χρησιμοποιηθεί σε μια εφαρμογή, για Ανάλυση Συναισθήματος σε κείμενο που θα εισάγει ο χρήστης διαδραστικά.

Τα αποτελέσματα που φαίνονται παρακάτω αντιστοιχούν για το μοντέλο BERT.

```
----------------------------------------------------------------------------------------------------
good for watching |---> neg
----------------------------------------------------------------------------------------------------
good for nothing |---> neg
----------------------------------------------------------------------------------------------------
This movie got me the chills from the beggining.Stunning visuals and good performance |---> pos
----------------------------------------------------------------------------------------------------
I hated this movie |---> neg
----------------------------------------------------------------------------------------------------
I have to do five projects at the same time.I guess u dont have to sleep every day. |---> neg
----------------------------------------------------------------------------------------------------
The film wasnt what i expected.In a bad way. |---> neg
----------------------------------------------------------------------------------------------------
I enjoyed this movie.Good production |---> pos
----------------------------------------------------------------------------------------------------
```

Επιπλέον, χρησιμοποιούμε ένα εργαλείο όπου μας δείχνει ποιες λέξεις του κειμένου είχαν τη μεγαλύτερη συνεισφορά για το τελικό αποτέλεσμα της κατηγοριοποίησης.

Οι χρωματισμένες λέξεις με πράσινο είναι αυτές με τη μεγαλύτερη συνεισφορά.

6. Συμπεράσματα

Στην παρούσα εργασία εξετάσαμε το πρόβλημα της Ανάλυσης Συναισθήματος, σε κριτικές ταινιών. Ο σκοπός ήταν η χρήση state-of-the-art αλγορίθμων για την αντιμετώπιση του προβλήματος.

Το καλύτερο μοντέλο είναι το BERT, αλλά απαιτεί αρκετό χρόνο εκπαίδευσης. Η συμπιεσμένη παραλλαγή του, το DistilBERT, για το συγκεκριμένο πρόβλημα, χρειάστηκε περίπου το 45% του χρόνου του BERT, πετυχαίνοντας το 98.2% της ακρίβειάς του και σε μια real-time εφαρμογή θα προτιμούσαμε αυτή την έκδοση. Το μοντέλο fastText αποδίδει καλά ως προς μνήμη και χρόνο, αλλά απαιτεί αρκετές επαναλήψεις, οι οποίες λόγω πρακτικών περιορισμών δεν ήταν δυνατό να γίνουν. Ο αλγόριθμος NBSVM μας εξέπληξε καθώς μόλις σε 1 λεπτό εκπαίδευσης αποδίδει ακρίβεια άνω του 90%.

Γενικά, η ανάλυση συναισθήματος είναι ένα πρόβλημα για το οποίο οι σύγχρονες τεχνικές αποδίδουν εξαιρετικά καλά. Το πρόβλημα που αντιμετωπίσαμε ήταν ο χρόνος εκπαίδευσης, δεδομένου ότι η εργασία έγινε σε σταθερό υπολογιστή. Χρησιμοποιώντας παραπάνω υπολογιστική ισχύ, θα μπορούσαμε να εκπαιδεύσουμε τα μοντέλα περεταίρω πετυχαίνοντας μεγαλύτερη ακρίβεια.

ΒΙΒΛΙΟΓΡΑΦΙΑ

[1] «Baselines and Bigrams: Simple, Good Sentiment and Topic Classification»
https://nlp.stanford.edu/pubs/sidaw12_simple_sentiment.pdf

[2] «Bag of Tricks for Efficient Text Classification»
https://arxiv.org/pdf/1607.01759.pdf#page=5&zoom=100,412,488

[3] «BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding»
https://arxiv.org/pdf/1810.04805.pdf

[4] DistilBERT: a distilled version of BERT: smaller, faster, cheaper and lighter https://arxiv.org/abs/1910.01108

[5] SGDR: Stochastic Gradient Descent with Warm Restarts
https://arxiv.org/abs/1608.03983

[6] Cyclical Learning Rates for Training Neural Networks
https://arxiv.org/abs/1506.01186