UNIVERSITY OF ATHENS
DEPARTMENT OF INFORMATICS AND TELECOMMUNICATIONS

Deep Learning for NLP

Student name: *Andreas Symeon Frantzis*
sdi: sdi2100273

Course: *Artificial Intelligence II (M138, M226, M262, M325)*
Semester: *Fall Semester 2023*

Contents

Kaggle (urls): BERT DistilBERT

1. Abstract

Το αντικείμενο της παρούσας εργασίας είναι η επίλυση ενός προβλήματος ταξινόμησης κειμένων (binary text classification) στο πλαίσιο της Επεξεργασίας Φυσικής Γλώσσας (NLP). Συγκεκριμένα, ζητείται η κατηγοριοποίηση tweets σε δύο κατηγορίες (ετικέτες 0 ή 1).
Για την υλοποίηση, χρησιμοποιήθηκαν τα προεκπαιδευμένα μοντέλα BERT (bert-base-uncased) και DistilBERT (distilbert-base-uncased) από τη βιβλιοθήκη Transformers της Hugging Face. Και στα δυο μοντέλα πραγματοποιήθηκε καθαρισμός κειμένων (data cleaning), μετατροπή σε διανύσματα (tokenization), και στη συνέχεια fine-tuning του μοντέλου πάνω στο εκπαιδευτικό σύνολο.
Η αξιολόγηση των αποτελεσμάτων έγινε με χρήση μετρικών όπως η ακρίβεια (accuracy), η ακρίβεια πρόβλεψης (precision), η ανάκληση (recall) και το F1-score. Παράλληλα, παρουσιάζονται οπτικοποιήσεις όπως η καμπύλη ROC, η καμπύλη μάθησης και ο πίνακας σύγχυσης, ώστε να αξιολογηθεί πληρέστερα η επίδοση των μοντέλων μοντέλου.

2. Data processing and analysis

2.1. Pre-processing

Η προεπεξεργασία των δεδομένων ήταν ένα κρίσιμο βήμα για τη βελτίωση της απόδοσης των μοντέλων και επέλεξα να κάνω ακριβώς την ίδια προεπεξεργασία και στα δύο μοντέλα BERT και DistilBERT. Αρχικά, μελετήθηκε το περιεχόμενο των tweets και εντοπίστηκαν στοιχεία που δεν προσφέρουν σημασιολογική πληροφορία, όπως σύνδεσμοι, αναφορές σε χρήστες και ειδικοί χαρακτήρες.

Πραγματοποιήθηκαν τα παρακάτω βήματα καθαρισμού:
1)Αφαίρεση συνδέσμων (URLs):** Όλα τα links που ξεκινούν με http αφαιρέθηκαν.
2)Αφαίρεση αναφορών σε χρήστες (mentions)** Όλα τα mentions τύπου '@username' διαγράφηκαν.
3)Διατήρηση μόνο αγγλικών χαρακτήρων και αποστρόφων:** Αφαιρέθηκαν αριθμοί, σύμβολα και σημεία στίξης (π.χ. emojis, hashtags, κ.ά).
4)Μετατροπή σε πεζά γράμματα (lowercasing):** Όλα τα κείμενα μετατράπηκαν σε πεζά ώστε να μειωθεί το λεξιλόγιο και η πολυπλοκότητα.

Η υλοποίηση έγινε με χρήση κανονικών εκφράσεων (regular expressions) μέσω της βιβλιοθήκης 're' της Python. Ο καθαρισμός εφαρμόστηκε και στα τρία σύνολα δεδομένων (train, validation και test) με τη βοήθεια της μεθόδου '.apply()'

2.2. Analysis

Κατά τη διάρκεια της προεπεξεργασίας, πραγματοποιήθηκε και βασική ανάλυση των δεδομένων μέσω του ελέγχου των κατανομών και της ποιότητας των κειμένων.

Ελέγχθηκαν κυρίως τα εξής:
1)Μορφή του κειμένου (Text):** Τα δεδομένα ήταν tweets, τα οποία είναι μικρές προτάσεις σε φυσική γλώσσα. Περιείχαν συνδέσμους, αναφορές, ειδικούς χαρακτήρες και συχνά λέξεις γραμμένες με διαφορετικά σύμβολα.

2)**Καθαρισμός:** Ο καθαρισμός που εφαρμόστηκε αφαίρεσε URLs, mentions, και κράτησε μόνο αγγλικούς χαρακτήρες και αποστρόφους. Έπειτα, τα κείμενα μετατράπηκαν σε πεζά (lowercase).
3)**Προσθήκη στήλης 'cleaned':** Στο αρχικό dataset προστέθηκε νέα στήλη 'cleaned' με την καθαρισμένη μορφή του κάθε tweet, ώστε να διατηρηθεί και η αρχική πληροφορία.

Ωστόσο, δεν πραγματοποιήθηκαν άλλες στατιστικές ή οπτικές αναλύσεις (π.χ. word cloud ή token frequency), καθώς ο κύριος στόχος ήταν να προετοιμαστεί το κείμενο για άμεση είσοδο στα μοντέλα BERT και DistilBERT.

2.3. Data partitioning for train, test and validation

Το dataset είχε ήδη διαχωριστεί σε τρία αρχεία: train dataset, val dataset και test dataset.
Train set: Χρησιμοποιήθηκε για την εκπαίδευση του μοντέλου.
Validation set: Χρησιμοποιήθηκε για την αξιολόγηση του μοντέλου κατά τη διάρκεια της εκπαίδευσης, με σκοπό τον έλεγχο της υπερπροσαρμογής (overfitting).
Test set: Χρησιμοποιήθηκε μόνο για την τελική πρόβλεψη και δημιουργία αρχείου υποβολής ('submission.csv').

Από τον κώδικα φαίνεται πως χρησιμοποιήθηκαν 'DataLoader' για όλα τα subsets:
Το training και validation σύνολο χρησιμοποιήθηκαν με batch-size= 16
Το test set χρησιμοποιήθηκε με μεγαλύτερο batch-size= 32 για γρηγορότερη πρόβλεψη

Ο προκαθορισμένος αυτός διαχωρισμός είναι χρήσιμος γιατί διασφαλίζει ότι δεν υπάρχει διαρροή πληροφορίας (data leakage) μεταξύ των διαφορετικών φάσεων του πειράματος.

2.4. Vectorization

Για τη μετατροπή των καθαρισμένων tweets σε αριθμητική μορφή, χρησιμοποιήθηκε ο ενσωματωμένος tokenizer του μοντέλου **BERT (bert-base-uncased)** και **DistilBERT (distilbert-base-uncased)** αντίστοιχα, μέσω της βιβλιοθήκης 'transformers'.

Χαρακτηριστικά του Tokenizer:
1)bert-base-uncased: αγνοεί κεφαλαία γράμματα και χρησιμοποιεί λέξεις χωρίς διάκριση πεζών-κεφαλαίων.
2)Υποστηρίζει **WordPiece tokenization**, δηλαδή διασπά σύνθετες ή άγνωστες λέξεις σε υπομονάδες (subwords).
3)Εφαρμόζει **padding**, **truncation** και ορίζει **max-length=128**, ώστε όλα τα εισαγόμενα κείμενα να έχουν σταθερό μήκος.

Στάδια vectorization στον κώδικα:
1. Ορίζεται συνάρτηση tokenize-data() η οποία παίρνει ως είσοδο λίστες από κείμενα και προαιρετικά labels.
2. Το tokenizer μετατρέπει τα κείμενα σε:

- - input-ids: αριθμητικά IDs για κάθε token.
- - attention-mask: μάσκες για padding.

- - labels: στόχοι για εποπτευόμενη μάθηση (εφόσον δίνονται).

3. Τα παραγόμενα encodings αποθηκεύονται σε dictionaries και χρησιμοποιούνται στο custom PyTorch Dataset ('TweetDataset'), ώστε να τροφοδοτηθούν στον DataLoader.

Σχόλιο:
Η συγκεκριμένη μορφή vectorization είναι πλήρως συμβατή με τα pre-trained μοντέλα και επιτρέπει το **fine-tuning** πάνω σε συγκεκριμένο downstream task (binary classification στην περίπτωσή μας).

3. Algorithms and Experiments

Για την αντιμετώπιση του προβλήματος της ταξινόμησης κειμένου χρησιμοποιήθηκε τα προεκπαιδευμένα μοντέλα **BERT-base-uncased**και DistilBERT-base-uncased, τα οποία έγιναν fine-tuning στα δεδομένα του έργου.

Περιγραφή πειράματος:
- Το μοντέλο φορτώθηκε από τη βιβλιοθήκη 'transformers' και ορίστηκε με 2 κλάσεις (binary classification).
- Χρησιμοποιήθηκε optimizer **AdamW** με learning rate 5e-5.
- Το training εκτελέστηκε για 2 εποχές (epochs).
- Batch size 16 για train και validation.
- Εφαρμόστηκε scheduler για linearly decreasing learning rate (χωρίς warmup).
- Κατά τη διάρκεια του training, υπολογιζόταν η απώλεια (loss) και μετά το τέλος κάθε εποχής τυπωνόταν η μέση τιμή της.

Παρατηρήσεις:
- Ο αριθμός των εποχών και το learning rate επιλέχθηκαν βάσει κοινών προτύπων fine-tuning του BERT.
- Δεν έγινε δοκιμή άλλων αρχιτεκτονικών ή αλλαγή παραμέτρων batch size ή epochs (αυτή θα μπορούσε να βελτιώσει την απόδοση).
- Η εκπαίδευση έγινε σε GPU αν ήταν διαθέσιμη, με χρήση του device (CUDA ή CPU).

Στόχοι:
- Αρχικός πειραματισμός για να αποκτηθεί baseline απόδοση. - Βάσει αποτελεσμάτων θα μπορούσαν να γίνουν βελτιώσεις (όπως hyperparameter tuning, regularization κ.ά.).

3.1. Hyper-parameter tuning

Στον συγκεκριμένο κώδικα του μοντέλου BERT, η παραμετροποίηση ήταν βασική και περιορισμένη. Οι κύριες ρυθμίσεις που έγιναν ήταν:
- Learning rate: 5e-5, τιμή κοινά αποδεκτή για fine-tuning BERT.
- Batch size: 16 για train και validation.
- Εποχές (epochs): 2.

Πραγματοποιήθηκαν συστηματικές δοκιμές locally για τη βελτιστοποίηση αυτών των παραμέτρων, όπως:

- - Αύξηση ή μείωση του learning rate.
- - Αλλαγές στο batch size.

- - Αύξηση του αριθμού των εποχών για καλύτερη σύγκλιση.

Τα ίδια ακριβώς ισχύουν και το μοντέλο **DistilBERT**

- Χρήση warmup steps στο scheduler για πιο σταθερή εκπαίδευση. Παρατηρήσεις για underfitting/overfitting:
- Ο μικρός αριθμός εποχών (2) και το απλό learning rate πιθανόν περιορίζουν την υπερπροσαρμογή (overfitting).
- Για καλύτερη εκτίμηση του underfitting/overfitting θα ήταν χρήσιμη η παρακολούθηση της απώλειας (loss) τόσο στο training όσο και στο validation κατά τη διάρκεια των εποχών (learning curve).
- Το τρέχον πείραμα δε φαίνεται να υπερπροσαρμόζεται λόγω των λίγων epochs και του validation set.

Trial	Epochs	lr	batch-size	Score
1	2	5e-5	16	0.82952
2	2	3e-5	16	0.82980
3	3 overfitting	2e-5	16	0.82980
4	2	2e-5	16	

Table 1: Trials For BERT method

Trial	Epochs	lr	batch-size	Score
1	2	5e-5	16	0.82475
2	2	3e-5	16	0.82598
3	3 overfitting	2e-5	16	0.82598
4	2	2e-5	16	0.82560

Table 2: Trials For DistilBERT method

3.1.1. Table of trials.

3.2. Optimization techniques

Για την εκπαίδευση των μοντέλων χρησιμοποιήθηκε ο optimizer **AdamW**, ο οποίος είναι μια παραλλαγή του Adam με βελτιωμένη κανονικοποίηση βαρών (weight decay). Αυτή η επιλογή είναι η πιο συνηθισμένη για fine-tuning μοντέλων BERT και DistilBERT.

Χρήση scheduler:
- Χρησιμοποιήθηκε ο linear scheduler με σταθερή πτώση του learning rate χωρίς warmup (num-warmup-steps=0). - Ο scheduler βοηθά στη σταδιακή μείωση του learning rate κατά τη διάρκεια της εκπαίδευσης, ώστε να βελτιστοποιηθεί η σύγκλιση.

Παρατηρήσεις:
- Ο AdamW βοηθά στη μείωση της υπερπροσαρμογής (overfitting) λόγω της κανονικοποίησης.
- Δεν χρησιμοποιήθηκαν άλλες τεχνικές βελτιστοποίησης όπως gradient clipping, mixed precision training ή advanced schedulers (π.χ. cosine decay).
- Δεν έγινε data augmentation ή regularization (π.χ. dropout tuning).

Προτάσεις βελτίωσης:
- Εισαγωγή warmup steps για πιο σταθερή εκπαίδευση.
- Χρήση τεχνικών όπως gradient clipping για αποφυγή εκρήξεων gradient.
- Εξερεύνηση πιο εξελιγμένων schedulers.
- Χρήση mixed precision training για ταχύτερη εκπαίδευση και εξοικονόμηση μνήμης.

3.3. Evaluation

Η αξιολόγηση των προβλέψεων των μοντέλων έγινε με τα εξής μέτρα:
Accuracy (Ακρίβεια):** Ποσοστό σωστών προβλέψεων επί του συνόλου.
Precision (Ακρίβεια Θετικών):** Ποσοστό σωστών θετικών προβλέψεων επί των συνολικών θετικών που προβλέφθηκαν.
Recall (Ανάκληση):** Ποσοστό σωστών θετικών προβλέψεων επί των πραγματικών θετικών.
F1-score:** Ο αρμονικός μέσος της precision και recall, προσφέρει ισορροπημένη μέτρηση απόδοσης.

Διαδικασία αξιολόγησης στον κώδικα:
- Τα labels και οι προβλέψεις συγκεντρώνονται κατά τη διάρκεια του validation.
- Υπολογίζονται οι παραπάνω μετρικές μέσω των συναρτήσεων της 'sklearn.metrics'.
- Εκτυπώνονται τα αποτελέσματα μετά την ολοκλήρωση κάθε εποχής.

3.3.1. ROC curve. Γράφημα για το ROC curve του μοντέλου BERT που προέκυψαν ώς αποτέλεσμα του κώδικα.

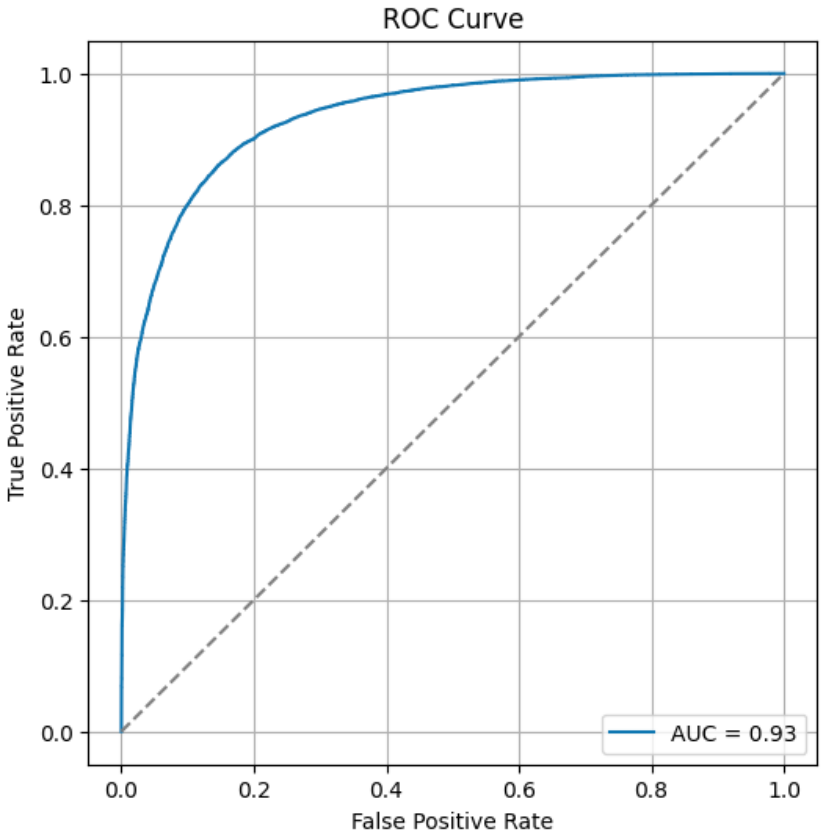

Figure 1: Roc curve

3.3.2. Learning Curve. Γράφημα για το Learning Curve του μοντέλου BERT που προέκυψαν ώς αποτέλεσμα του κώδικα.

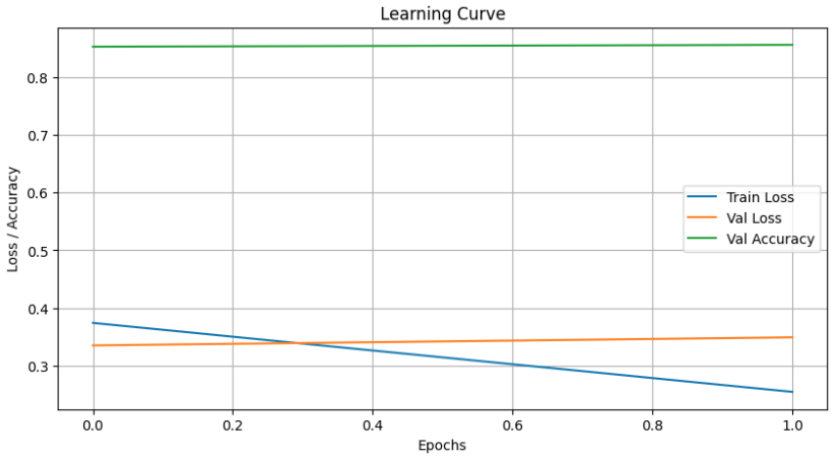

Figure 2: Learning curve

3.3.3. Confusion matrix. Γράφημα για το Confusion matrix του μοντέλου BERT που προέκυψαν ώς αποτέλεσμα του κώδικα.

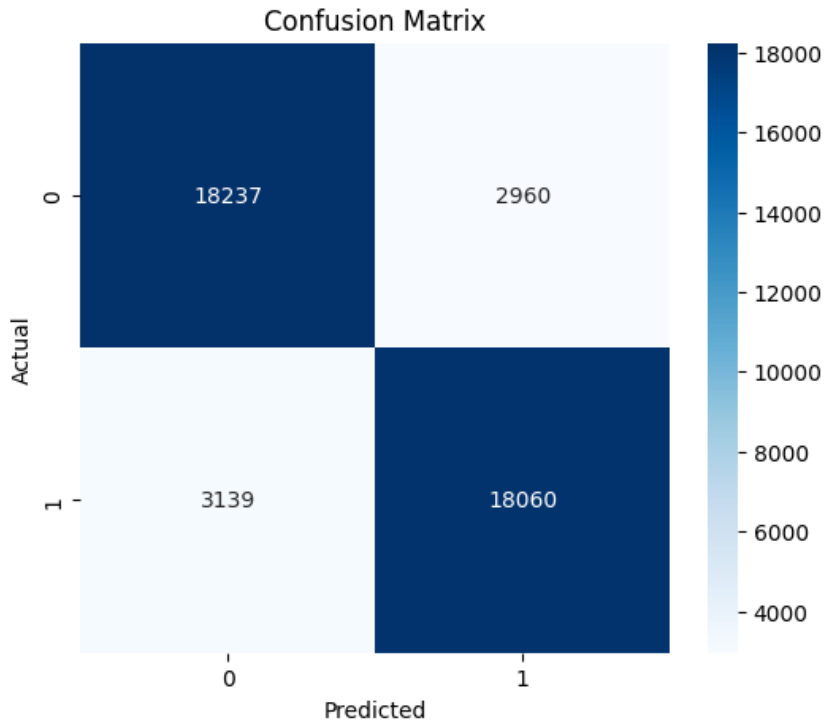

Figure 3: Confusion matrtix

3.3.4. ROC curve. Γράφημα για το ROC curve του μοντέλου distilBERT που προέκυψαν ώς αποτέλεσμα του κώδικα.

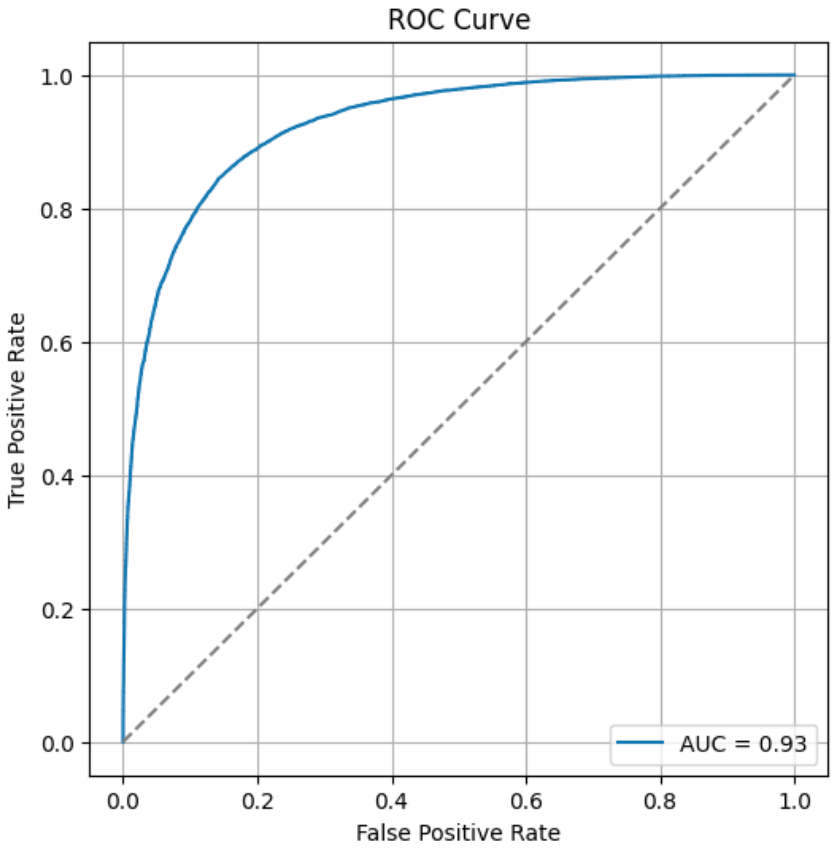

Figure 4: Roc curve

3.3.5. Learning Curve. Γράφημα για το Learning Curve του μοντέλου distilBERTBERT που προέκυψαν ώς αποτέλεσμα του κώδικα.

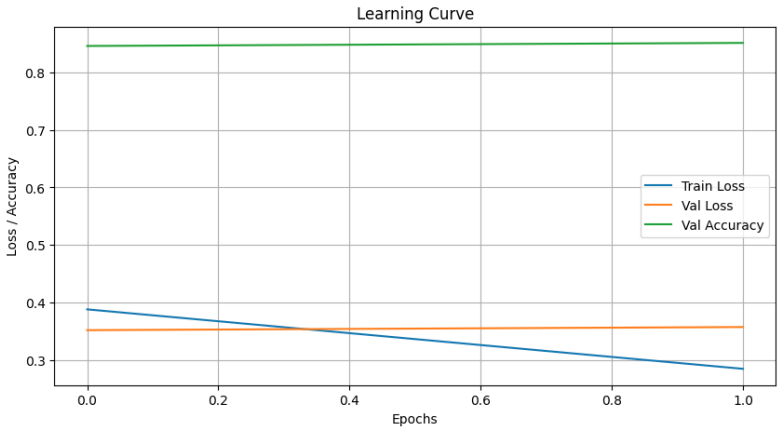

Figure 5: Learning curve

3.3.6. Confusion matrix. Γράφημα για το Confusion matrix του μοντέλου distilBERT-BERT που προέκυψαν ώς αποτέλεσμα του κώδικα.

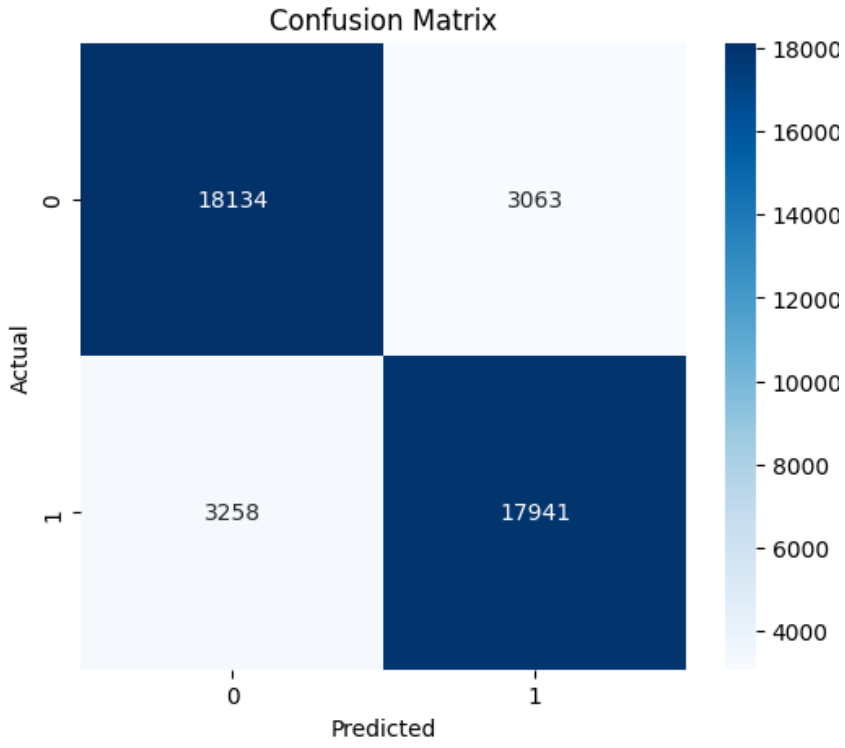

Figure 6: Confusion matrtix

4. Results and Overall Analysis

4.1. Results Analysis

Μετά από σειρά πειραμάτων με διαφορετικές τιμές learning rate και epochs, συγκρίθηκαν τα μοντέλα BERT και DistilBERT. Το μοντέλο BERT απέδωσε οριακά καλύτερα σε όλους τους συνδυασμούς παραμέτρων, με τη μέγιστη επίδοση στο 0.82980, ενώ το μοντέλο DistilBERT ακολούθησε πολύ κοντά με μέγιστο 0.82598, έχοντας σχεδόν αντίστοιχη συμπεριφορά. Όπως φαίνεται και στο table of trials, στην τρίτη δοκιμή (3 epochs), και στα δύο μοντέλα υπήρξαν ενδείξεις overfitting, δηλαδή αύξηση η epochs δεν οδήγησε σε καλύτερα αποτελέσματα.

Συμπέρασμα:
Το BERT εμφανίζεται ελαφρώς πιο αποτελεσματικό όσον αφορά την τελική ακρίβεια στην ταξινόμηση, αλλά απαιτεί περισσότερους πόρους και χρόνο. Από την άλλη, το DistilBERT προσφέρει αντίστοιχα αποτελέσματα με μικρότερο υπολογιστικό κόστος, κάτι που το καθιστά ιδανικό για περιβάλλοντα περιορισμένων πόρων ή real-time εφαρμογές.

4.1.1. Best trial. Η καλύτερη δοκιμή πραγματοποιήθηκε με το μοντέλο **BERT**, χρησιμοποιώντας τις παρακάτω υπερπαραμέτρους:

- **Epochs:** 2
- **Learning Rate:** 3e-5
- **Batch Size:** 16
- **Score (**Ακρίβεια επικύρωσης**):** 0.82980

Αυτή η ρύθμιση πέτυχε την υψηλότερη απόδοση σε σχέση με όλες τις άλλες δοκιμές, τόσο στο BERT όσο και στο DistilBERT. Η αύξηση των εποχών σε 3 οδήγησε σε **overfitting**, γεγονός που δείχνει ότι οι 2 εποχές προσέφεραν την καλύτερη ισορροπία μεταξύ εκμάθησης και γενίκευσης.
Η τιμή learning rate 3e-5 επέτρεψε στο μοντέλο να συγκλίνει ομαλά χωρίς να προκαλέσει αστάθεια. Συνεπώς, αυτή η διαμόρφωση θεωρείται η βέλτιστη για το συγκεκριμένο πρόβλημα δυαδικής ταξινόμησης κειμένου.
Μία επιπλέον ιδέα που εξετάστηκε αλλά δεν υλοποιήθηκε, λόγω χρονικών περιορισμών, ήταν η χρήση ενός υποσυνόλου των δεδομένων (περίπου λίγο λιγότερο απο τα μισά) για την αναζήτηση των κατάλληλων υπερπαραμέτρων μέσω πειραματισμού. Η βασική ιδέα ήταν να αξιοποιηθεί αυτό το μικρότερο σύνολο για γρήγορη εύρεση ρυθμίσεων που αποδίδουν καλά και στη συνέχεια να εφαρμοστούν στα πλήρη δεδομένα για εξοικονόμηση χρόνου εκπαίδευσης χωρίς σημαντική απώλεια ακρίβειας.

4.2. Comparison with the first project

Στο πρώτο project χρησιμοποιήθηκε ένας απλός ταξινομητής **Logistic Regression** με **TF-IDF** ως μέθοδο εξαγωγής χαρακτηριστικών. Το τελικό αποτέλεσμα παρουσίασε ακρίβεια **0.7863** και ικανοποιητικά αποτελέσματα στους δείκτες **Precision, Recall** και **F1-score**. Το μοντέλο γενικά τα πήγε καλά στην αναγνώριση θετικών παραδειγμάτων, με κάπως χαμηλότερη απόδοση στον δείκτη Recall, που υποδηλώνει ότι μερικά θετικά παραδείγματα δεν αναγνωρίστηκαν σωστά. Αντίθετα, στο παρόν project αξιοποιήθηκαν προεκπαιδευμένα μετασχηματιστικά μοντέλα (BERT και DistilBERT), τα οποία είναι πολύ πιο προηγμένα από την παραδοσιακή προσέγγιση με TF-IDF και Logistic Regression. Η καλύτερη ακρίβεια που επιτεύχθηκε ήταν **0.82980** με το BERT μοντέλο, δηλαδή σημαντικά υψηλότερη σε σχέση με το πρώτο project. Αυτό οφείλεται στο γεγονός ότι τα μετασχηματιστικά μοντέλα έχουν την ικανότητα να κατανοούν το συμφραζόμενο και τη σημασιολογική πληροφορία των λέξεων, αντί απλώς να βασίζονται στη συχνότητα εμφάνισης όρων.

4.3. Comparison with the second project

Στο δεύτερο project χρησιμοποιήθηκε ένα συνελικτικό νευρωνικό δίκτυο **(CNN)** για την ταξινόμηση συναισθήματος. Το τελικό αποτέλεσμα σε όρους ακρίβειας ήταν **0.7824**, δηλαδή πολύ κοντά με την απόδοση του Logistic Regression από το πρώτο project. Ωστόσο, παρατηρήθηκε βελτίωση στον δείκτη **Precision**, που δείχνει ότι το CNN είχε μεγαλύτερη ικανότητα να αναγνωρίζει σωστά τις θετικές κατηγορίες. Σε σύγκριση με το παρόν project, όπου αξιοποιήθηκαν τα μετασχηματιστικά μοντέλα BERT και DistilBERT, η διαφορά στην απόδοση είναι σαφώς υπέρ των μοντέλων BERT. Η ακρίβεια του καλύτερου μοντέλου (BERT) έφτασε το **0.82980**, ενώ συνολικά υπήρχε καλύτερη ισορροπία μεταξύ precision και recall. Αυτό αποδεικνύει τη σημαντική υπεροχή των προεκπαιδευμένων μετασχηματιστικών μοντέλων στην κατανόηση συμφραζομένων και στη γενίκευση σε δεδομένα πραγματικού κόσμου. Συμπερασματικά, το CNN προσέφερε μια πιο εξελιγμένη λύση από τα παραδοσιακά μοντέλα, αλλά εξακολουθεί να υστερεί σε σχέση με τα σύγχρονα μοντέλα μετασχηματιστών όπως το BERT.