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

1. Abstract

Ο στόχος αυτής της εργασίας είναι η ανάπτυξη ενός ταξινομητή συναισθήματος (sentiment classifier) για αγγλόφωνα tweets. Κάθε tweet συνοδεύεται από μια δυαδική ετικέτα: θετικό ή ουδέτερο συναίσθημα (κλάση 0) ή αρνητικό συναίσθημα (κλάση 1).

Για την επίλυση του προβλήματος, ακολουθήσαμε την εξής προσέγγιση:

- Φόρτωση και Προεπεξεργασία Δεδομένων**:** Φορτώνουμε τα δεδομένα εκπαίδευσης, επικύρωσης και δοκιμής από αρχεία CSV. Το κείμενο καθαρίζεται μέσω αφαίρεσης URLs και αναφορών σε χρήστες (@mentions), μετατροπής σε πεζά και tokenization με χρήση του `nltk`.
- Αναπαράσταση Κειμένου με **Word2Vec:** Εκπαιδεύουμε μοντέλο Word2Vec πάνω στα tokenized tweets του training set χρησιμοποιώντας skip-gram και διαστάσεις 100. Οι παραγόμενες λέξεις-διανύσματα (embeddings) χρησιμοποιούνται για αρχικοποίηση του embedding layer του μοντέλου.
- Αρχιτεκτονική Μοντέλου**:** Ο ταξινομητής αποτελείται από ένα embedding layer (μη παγωμένο), μία κρυφή πλήρως συνδεδεμένη στρώση με ReLU ενεργοποίηση, dropout για regularization, και μία τελική έξοδο δύο διαστάσεων (softmax).
- Εκπαίδευση και Βελτιστοποίηση**:** Το μοντέλο εκπαιδεύεται με χρήση της συνάρτησης κόστους cross-entropy και του βελτιστοποιητή Adam. Οι υπερπαράμετροι (μέγεθος dropout, learning rate, διαστάσεις κρυφής στρώσης) επιλέγονται μέσω βελτιστοποίησης με Optuna.
- Αξιολόγηση και Υποβολή**:** Το τελικό μοντέλο αξιολογείται σε validation set ως προς ακρίβεια και F_1 score, και γίνεται πρόβλεψη στο test set για δημιουργία αρχείου υποβολής.

2. Data processing and analysis

2.1. Pre-processing

Στο πλαίσιο της παρούσας εργασίας, εφαρμόστηκαν τα παρακάτω βήματα καθαρισμού και μετασχηματισμού των κειμένων των tweets:

- Αφαίρεση **URL:** Με χρήση κανονικών εκφράσεων αφαιρέθηκαν όλα τα URLs (π.χ. `http://...`), καθώς δεν παρέχουν χρήσιμη πληροφορία για το συναισθηματικό περιεχόμενο του tweet.
- Αφαίρεση αναφορών σε χρήστες **(@mentions):** Οι αναφορές σε άλλους χρήστες (π.χ. `@username`) δεν συμβάλλουν στη διάκριση του συναισθήματος και αφαιρέθηκαν επίσης.
- Καθαρισμός χαρακτήρων**:** Κρατήθηκαν μόνο αγγλικοί χαρακτήρες και απόστροφοι. Όλοι οι υπόλοιποι χαρακτήρες (σύμβολα, αριθμοί, σημεία στίξης) αφαιρέθηκαν.
- Μετατροπή σε πεζά**:** Όλο το κείμενο μετατράπηκε σε πεζά γράμματα για ομοιομορφία (case folding).
- **Tokenization:** Το καθαρισμένο κείμενο μετατράπηκε σε λίστα λέξεων (tokens) με χρήση της `word_tokenize()` από τη βιβλιοθήκη `nltk`.

- Διατήρηση λέξεων με ελάχιστη συχνότητα**:** Κατά την εκπαίδευση του Word2Vec, αγνοήθηκαν λέξεις που εμφανίζονται λιγότερες από 2 φορές στο σύνολο του training set, για να περιοριστεί ο θόρυβος από σπάνιες ή ανορθόγραφες λέξεις.

- Μέγιστο μήκος **tweet:** Κατά τη μετατροπή των tweets σε αριθμητικά διανύσματα, διατηρήθηκε μόνο το πρώτο μέχρι **30 tokens** ανά tweet. Τα sequences μικρότερου μήκους γέμιζαν με padding.

Η παραπάνω διαδικασία βοήθησε στη δημιουργία ενός καθαρού και συνεκτικού συνόλου δεδομένων, το οποίο είναι κατάλληλο για εκπαίδευση νευρωνικών δικτύων.

2.2. Analysis

Παράλληλα με την προεπεξεργασία, πραγματοποιήθηκε ανάλυση των δεδομένων για να κατανοηθεί καλύτερα η φύση τους και να ληφθούν αποφάσεις για την κατάλληλη αρχιτεκτονική του μοντέλου.

- Μήκος **tweets:** Η κατανομή του πλήθους των λέξεων ανά tweet μελετήθηκε για να επιλεγεί κατάλληλο μέγιστο μήκος ακολουθίας. Παρατηρήθηκε ότι η πλειονότητα των tweets περιέχει λιγότερες από 30 λέξεις, επομένως επιλέχθηκε ως μέγιστο μήκος η τιμή των 30 tokens.

- Συχνότητες λέξεων**:** Πραγματοποιήθηκε υπολογισμός της συχνότητας εμφάνισης των λέξεων σε όλο το training set. Οι πιο συχνές λέξεις ήταν λέξεις όπως “i", “the", “to", “my", “and", οι οποίες είναι κοινές στο αγγλικό λεξιλόγιο και δεν φέρουν ισχυρή συναισθηματική πληροφορία. Παρόλα αυτά, δεν εφαρμόστηκε stopword removal ώστε να μην αφαιρεθούν ενδεχομένως χρήσιμες συναισθηματικά λέξεις.

- Στατιστική κατανομή ετικετών**:** Οι ετικέτες (Label) ήταν ισορροπημένες με περίπου ίσο αριθμό παραδειγμάτων για τις κατηγορίες 0 και 1, γεγονός που καθιστά την ακρίβεια (accuracy) αξιόπιστο μέτρο αξιολόγησης.

- Μέγεθος λεξιλογίου**:** Το λεξιλόγιο που προέκυψε από την εκπαίδευση του Word2Vec αποτελείται από περίπου 27.351 διαφορετικές λέξεις.

Η παραπάνω ανάλυση επέτρεψε την κατανόηση της φύσης των δεδομένων και βοήθησε στη λήψη τεκμηριωμένων αποφάσεων για την κατασκευή του μοντέλου.

2.3. Data partitioning for train, test and validation

Το αρχικό σύνολο δεδομένων διαχωρίστηκε σε τρία υποσύνολα: εκπαίδευσης (training), επικύρωσης (validation) και δοκιμής (test), με αναλογία περίπου 70%-15%-15%. Συγκεκριμένα:

- **Training set (70%):** Χρησιμοποιείται για την εκπαίδευση του μοντέλου. Περιλαμβάνει το μεγαλύτερο μέρος των δεδομένων ώστε το μοντέλο να μάθει τα πρότυπα που υπάρχουν στα tweets.

- **Validation set (15%):** Χρησιμοποιείται για την επιλογή των υπερπαραμέτρων του μοντέλου (όπως learning rate, batch size κ.ά.) και για να παρακολουθείται η απόδοση κατά τη διάρκεια της εκπαίδευσης. Επιτρέπει την αποφυγή υπερπροσαρμογής (overfitting).

- **Test set (15%):** Χρησιμοποιείται μόνο για την τελική αξιολόγηση του μοντέλου, αφού έχει ολοκληρωθεί η εκπαίδευση. Παρέχει μια αντικειμενική μέτρηση της γενίκευσης του μοντέλου σε άγνωστα δεδομένα.

Ο διαχωρισμός έγινε με τυχαία ανάμειξη (random shuffle) των δεδομένων, διασφαλίζοντας ότι το κάθε υποσύνολο έχει παρόμοια κατανομή ως προς τις ετικέτες (stratified split). Αυτή η μεθοδολογία θεωρείται καλή πρακτική στη μηχανική μάθηση και εξασφαλίζει αξιόπιστη αξιολόγηση του μοντέλου.

2.4. Vectorization

Για την αναπαράσταση των λέξεων των tweets ως διανύσματα, χρησιμοποιήθηκε η τεχνική **Word2Vec**, η οποία είναι μια τεχνική υπολογισμού ενσωματώσεων (embeddings) λέξεων. Η τεχνική αυτή μετατρέπει τις λέξεις σε διανύσματα συνεχών τιμών που καταγράφουν τη σημασιολογία της κάθε λέξης με βάση το συμφραζόμενο (context) στο οποίο εμφανίζεται.

- **Word2Vec:** Χρησιμοποιήθηκε το μοντέλο `Word2Vec` από τη βιβλιοθήκη `gensim`. Το μοντέλο αυτό εκπαιδεύτηκε πάνω στα δεδομένα των tweets, και τα διανύσματα (embeddings) λέξεων που προέκυψαν έχουν διάσταση 100. Αυτά τα διανύσματα αναπαριστούν τις λέξεις σε έναν πολυδιάστατο χώρο και επιτρέπουν στο μοντέλο να κατανοήσει τις σχέσεις μεταξύ των λέξεων.

- **Skip-Gram Model:** Επιλέχθηκε το μοντέλο Skip-Gram, το οποίο προβλέπει το συμφραζόμενο (context) μιας λέξης βασισμένο στην ίδια την λέξη. Το Skip-Gram είναι κατάλληλο για την ανίχνευση σπάνιων λέξεων και παρέχει καλές ενσωματώσεις λέξεων όταν το μέγεθος του κειμένου είναι μεγάλο.

- Εκπαίδευση του **Word2Vec:** Το μοντέλο `Word2Vec` εκπαιδεύτηκε με τα tweets, χρησιμοποιώντας παράμετρο `window=5`, που καθορίζει τον αριθμό των γειτονικών λέξεων γύρω από την κάθε λέξη που θα εξετάζονται κατά την εκπαίδευση.

- Δημιουργία του **Embedding Matrix:** Μετά την εκπαίδευση του μοντέλου Word2Vec, δημιουργήθηκε ένας πίνακας ενσωμάτωσης (embedding matrix), ο οποίος περιέχει τα διανύσματα όλων των λέξεων του λεξιλογίου που εντοπίστηκαν κατά την εκπαίδευση. Για τις λέξεις που δεν εμφανίζονται στο λεξιλόγιο, χρησιμοποιήθηκε η τυχαία αρχικοποίηση τους με κανονική κατανομή. Επιπλέον, προστέθηκαν ειδικά διανύσματα για το padding (στοιχεία που προστίθενται για την ομογενοποίηση του μήκους των ακολουθιών) και για τις άγνωστες λέξεις (UNK).

- Χρήση του **Embedding Layer** στο Μοντέλο**:** Το embedding matrix ενσωματώθηκε στον αρχικό επιπέδο του μοντέλου ως ένα `Embedding Layer` στο PyTorch, το οποίο μετατρέπει κάθε λέξη σε ένα αντίστοιχο διάνυσμα κατά τη διάρκεια της εκπαίδευσης.

Αυτή η τεχνική επιτρέπει στο μοντέλο να κατανοήσει τη σημασία κάθε λέξης, βασισμένο στο συμφραζόμενο και να ενσωματώσει την πληροφορία αυτή σε ένα συνεχές και πυκνό χώρο χαρακτηριστικών.

3. Algorithms and Experiments

3.1. Experiments

Για τη βελτιστοποίηση των υπερπαραμέτρων του μοντέλου, χρησιμοποίησα την βιβλιοθήκη Optuna, η οποία πραγματοποιεί τυχαία ή στοχευμένη αναζήτηση στον χώρο των υπερπαραμέτρων για να βρει τον καλύτερο συνδυασμό. Η διαδικασία βελτιστοποίησης περιλάμβανε τα εξής βήματα:

- Επιλογή υπερπαραμέτρων προς βελτιστοποίηση:
  - `hidden_dim`: Διαστάσεις του κρυμμένου επιπέδου (64 έως 512).
  - `lr`: Ρυθμός μάθησης (από `1e-5` έως `1e-1` σε λογαριθμική κλίμακα).
  - `dropout`: Πιθανότητα dropout (0.1 έως 0.5).
- Χρήση του Optuna για την αναζήτηση του καλύτερου συνδυασμού υπερπαραμέτρων μέσω των πειραμάτων.
- Εκπαίδευση του μοντέλου για 5 εποχές και αξιολόγηση με χρήση του F1-score στην επικύρωση.

Ο κώδικας που χρησιμοποιήθηκε είναι ο εξής:

```
import optuna
device = torch.device("cuda" if torch.cuda.is_available() else "cpu")

def objective(trial):
    hidden_dim = trial.suggest_int("hidden_dim", 64, 512)
    lr = trial.suggest_float("lr", 1e-5, 1e-1, log=True)
    dropout = trial.suggest_float("dropout", 0.1, 0.5)

    model = SentimentClassifier(embedding_matrix, hidden_dim, dropout).to(device)
    optimizer = optim.Adam(model.parameters(), lr=lr)
    criterion = nn.CrossEntropyLoss()

    for epoch in range(5):
        model.train()
        for batch_x, batch_y in train_loader:
            batch_x, batch_y = batch_x.to(device), batch_y.to(device)
            optimizer.zero_grad()
            logits = model(batch_x)
            loss = criterion(logits, batch_y)
            loss.backward()
            optimizer.step()

    metrics = evaluate(model, val_loader)
    return metrics["f1"]

study = optuna.create_study(direction="maximize")
```

```
study.optimize(objective, n_trials=2)

print("Best hyperparameters:", study.best_trial.params)
```

Μετά από 2 πειράματα, οι καλύτερες υπερπαράμετροι που βρέθηκαν ήταν οι εξής:

- `hidden_dim = 442`
- `lr = 6.7e-5`
- `dropout = 0.48`

Αυτές οι παράμετροι οδήγησαν στην καλύτερη απόδοση του μοντέλου στην αξιολόγηση της επικύρωσης.

3.2. Hyper-parameter tuning

Μετά την εκτέλεση της βελτιστοποίησης υπερπαραμέτρων μέσω του Optuna, το τελικό μοντέλο διαμορφώθηκε με βάση τις καλύτερες υπερπαράμετροι που προέκυψαν από την αναζήτηση. Οι καλύτερες υπερπαράμετροι για το μοντέλο ήταν:

- `hidden_dim` = 442: Διαστάσεις του κρυμμένου επιπέδου.
- `lr` = 6.7e-5: Ρυθμός μάθησης.
- `dropout` = 0.48: Πιθανότητα dropout.

Η εκπαίδευση του μοντέλου πραγματοποιήθηκε για 5 εποχές, με τη χρήση του Adam optimizer και της συνάρτησης κόστους `CrossEntropyLoss`.

3.3. Διαμόρφωση του Μοντέλου

Το μοντέλο που χρησιμοποιήθηκε ήταν ένα απλό feedforward νευρωνικό δίκτυο (MLP), το οποίο αποτελούνταν από:

- Ένα επίπεδο εισόδου που περιείχε τα Word2Vec embeddings των λέξεων.
- Ένα ή περισσότερα κρυμμένα επίπεδα, με `hidden_dim` μονάδες.
- Ένα επίπεδο εξόδου με δύο κλάσεις (θετική και αρνητική διάθεση).
- Χρήση dropout για να αποφευχθεί η υπερπροσαρμογή του μοντέλου στα δεδομένα εκπαίδευσης.

Η εκπαίδευση πραγματοποιήθηκε για 5 εποχές και το μοντέλο αξιολογήθηκε χρησιμοποιώντας το F1-score στην επικύρωση.

3.4. Υποπροβλήματα Υπερβολικής/Ελλιπούς Προσαρμογής **(Underfitting/Overfitting)**

Κατά την εκπαίδευση και αξιολόγηση του μοντέλου, παρατηρήθηκαν οι εξής περιπτώσεις όσον αφορά την υπερπροσαρμογή και την ελλιπή προσαρμογή:

- Υπερπροσαρμογή **(Overfitting):** Σε περιπτώσεις όπου το μοντέλο δεν χρησιμοποιούσε dropout ή η τιμή του dropout ήταν πολύ χαμηλή, παρατηρήθηκε ότι το μοντέλο υπερεκπαιδευόταν στα δεδομένα εκπαίδευσης και είχε χαμηλή απόδοση στην επικύρωση. Για να αντιμετωπιστεί η υπερπροσαρμογή, αυξήσαμε την τιμή του dropout σε 0.48 και χρησιμοποιήσαμε κανονικοποίηση μέσω του Adam optimizer.
- Ελλιπής Προσαρμογή **(Underfitting):** Στην περίπτωση που οι υπερπαράμετροι του μοντέλου ήταν πολύ μικρές (π.χ., χαμηλός αριθμός κρυφών μονάδων ή πολύ χαμηλός ρυθμός μάθησης), το μοντέλο δεν κατάφερε να μάθει πλήρως τα δεδομένα, με αποτέλεσμα την ελλιπή προσαρμογή. Το μοντέλο είχε χαμηλά αποτελέσματα στην εκπαίδευση και στην επικύρωση. Αυτή η περίπτωση επιλύθηκε με την αύξηση του αριθμού των κρυφών μονάδων (π.χ., 442 μονάδες) και του ρυθμού μάθησης.

3.5. Αξιολόγηση των Αποτελεσμάτων

Η απόδοση του μοντέλου αξιολογήθηκε χρησιμοποιώντας διάφορους δείκτες απόδοσης όπως το F1-score και ο ρυθμός ακριβείας. Στη συνέχεια, για τη σύγκριση της απόδοσης του μοντέλου, παραθέτουμε τις καμπύλες εκμάθησης (learning curves) για την εκπαίδευση και την επικύρωση. Η γραφική αναπαράσταση δείχνει ότι η απόδοση του μοντέλου σταθεροποιείται μετά από τις πρώτες 2-3 εποχές.

Επιπλέον, οι πίνακες σύγχυσης (confusion matrices) αποτυπώνουν την απόδοση του μοντέλου στις κατηγορίες, δείχνοντας τα ποσοστά σωστών και λανθασμένων ταξινομήσεων για κάθε κατηγορία (θετική/αρνητική διάθεση).

3.6. Optimization techniques

Στη διαδικασία εκπαίδευσης του μοντέλου, χρησιμοποιήσαμε διάφορες τεχνικές και πλαίσια βελτιστοποίησης για να επιτύχουμε την καλύτερη απόδοση. Εδώ περιγράφουμε τις βασικές τεχνικές που χρησιμοποιήθηκαν κατά τη διάρκεια των πειραμάτων μας.

3.7. Optuna για Βελτιστοποίηση Υπερπαραμέτρων

Για τη βελτιστοποίηση των υπερπαραμέτρων του μοντέλου, χρησιμοποιήσαμε το πλαίσιο `Optuna`. Το `Optuna` είναι ένα πλαίσιο αυτόματης βελτιστοποίησης υπερπαραμέτρων, που επιτρέπει τη δημιουργία πειραμάτων και την εύρεση των καλύτερων υπερπαραμέτρων για το μοντέλο.

3.7.1. Βασικά Στοιχεία του ***Optuna.*** Η βασική διαδικασία περιλάμβανε την οριστικοποίηση των υπερπαραμέτρων του μοντέλου, όπως:

- `hidden_dim`: Ο αριθμός των νευρώνων στο κρυφό επίπεδο.
- `lr`: Ο ρυθμός μάθησης.
- `dropout`: Η πιθανότητα απόρριψης μονάδων κατά τη διάρκεια της εκπαίδευσης για την αποφυγή υπερπροσαρμογής.

Για κάθε πείραμα, το `Optuna` προσπάθησε να βρει τις καλύτερες τιμές αυτών των παραμέτρων και να μεγιστοποιήσει το `F1-score`, που χρησιμοποιήθηκε ως μέτρο αξιολόγησης της απόδοσης του μοντέλου. Ο αριθμός των δοκιμών (trials) που εκτελέστηκαν ήταν 2, ώστε να εξετάσουμε τα πρώτα αποτελέσματα, αν και σε πραγματικές συνθήκες θα εκτελούνταν περισσότερες δοκιμές για να βελτιώσουμε περαιτέρω την απόδοση.

3.8. Επιλογή Αλγορίθμου Βελτιστοποίησης

Για την εκπαίδευση του μοντέλου, χρησιμοποιήσαμε τον αλγόριθμο βελτιστοποίησης `Adam`, ο οποίος είναι ένας από τους πιο διαδεδομένους αλγορίθμους για την εκπαίδευση βαθιών νευρωνικών δικτύων. Ο `Adam` είναι ένας αλγόριθμος που συνδυάζει τα πλεονεκτήματα του `Momentum` και της `RMSprop`, επιτρέποντας τη γρήγορη και αποδοτική σύγκλιση κατά τη διάρκεια της εκπαίδευσης. Η χρήση του `Adam` μας επέτρεψε να διατηρήσουμε έναν σχετικά μικρό ρυθμό μάθησης και να αποφύγουμε τα προβλήματα υπερβολικής μεταβολής του κόστους κατά την εκπαίδευση.

3.9. Ρύθμιση Υπερπαραμέτρων

Για τη βελτιστοποίηση του ρυθμού μάθησης (`lr`) και της πιθανότητας dropout (`dropout`), πραγματοποιήθηκαν πειράματα σε διάφορες τιμές, με σκοπό να βρούμε τις βέλτιστες ρυθμίσεις. Ο ρυθμός μάθησης ρυθμίστηκε σε τιμές που κυμαίνονταν από 10^{-5} έως 10^{-1} και η πιθανότητα dropout κυμαίνονταν μεταξύ 0.1 και 0.5. Οι τιμές αυτές επηρεάζουν την ταχύτητα σύγκλισης και την ικανότητα του μοντέλου να αποφύγει την υπερπροσαρμογή.

3.10. Στρατηγικές Κανονικοποίησης και Αποφυγής Υπερπροσαρμογής

Εκτός από τη χρήση του dropout, ο `Adam` optimizer μας παρείχε μια μορφή κανονικοποίησης, ελαχιστοποιώντας το κόστος και παρέχοντας καλύτερη γενίκευση. Επιπλέον, πραγματοποιήθηκαν πειράματα με την αύξηση του αριθμού των εποχών και τη χρήση νωρίτερης διακοπής της εκπαίδευσης (early stopping) για την αποφυγή υπερπροσαρμογής και τη βελτίωση της απόδοσης στο σύνολο επικύρωσης.

3.11. Αξιολόγηση της Απόδοσης του Μοντέλου

Κατά τη διάρκεια της εκπαίδευσης, παρακολουθήσαμε την εξέλιξη της απόδοσης του μοντέλου μέσω διαγραμμάτων μάθησης (learning curves) για την εκπαίδευση και την επικύρωση. Αυτά τα διαγράμματα μας επέτρεψαν να παρατηρήσουμε αν υπήρχε πρόβλημα υπερπροσαρμογής ή ελλιπούς προσαρμογής, και να προσαρμόσουμε ανάλογα τις υπερπαραμέτρους.

Επιπλέον, χρησιμοποιήσαμε πίνακες σύγχυσης (confusion matrices) για να κατανοήσουμε καλύτερα πώς ταξινομούσε το μοντέλο τις διάφορες κλάσεις και αν υπήρχαν σφάλματα στη διάκριση των θετικών και αρνητικών συναισθημάτων.

3.12. Evaluation

Για την αξιολόγηση των προβλέψεων του μοντέλου, χρησιμοποιούμε διάφορες μετρικές που μας επιτρέπουν να κατανοήσουμε την απόδοση του μοντέλου. Οι πιο συχνά χρησιμοποιούμενες μετρικές είναι οι εξής: Ακρίβεια **(Accuracy)**, Ακρίβεια Θετικών **(Precision)**, Ανάκληση **(Recall)**, **F1 Score** και ο Πίνακας Σύγχυσης **(Confusion Matrix)**.

1. Ακρίβεια **(Accuracy)**

Η ακρίβεια μετρά το ποσοστό των σωστών προβλέψεων σε σχέση με το σύνολο των παρατηρήσεων. Υπολογίζεται ως εξής:

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

Όπου:

- TP: Σωστά Θετικά (True Positives)
- TN: Σωστά Αρνητικά (True Negatives)
- FP: Λανθασμένα Θετικά (False Positives)
- FN: Λανθασμένα Αρνητικά (False Negatives)

2. Ακρίβεια Θετικών **(Precision)**

Η ακρίβεια θετικών (ή θετική ακρίβεια) δείχνει το ποσοστό των θετικών προβλέψεων που ήταν σωστές. Η φόρμουλα είναι:

$$\text{Ακρίβεια Θετικών} = \frac{TP}{TP + FP}$$

3. Ανάκληση **(Recall)**

Η ανάκληση (ή ευαισθησία) δείχνει το ποσοστό των πραγματικών θετικών που το μοντέλο κατάφερε να εντοπίσει σωστά. Υπολογίζεται ως εξής:

$$\text{Ανάκληση} = \frac{TP}{TP + FN}$$

4. F1 Score

Το **F1 Score** είναι ο αρμονικός μέσος όρος της ακρίβειας και της ανάκλησης και χρησιμοποιείται για να συνδυάσει την ακρίβεια και την ανάκληση σε ένα μόνο μέτρο. Ο υπολογισμός του είναι:

$$\text{F1 Score} = 2 \times \frac{\text{Ακρίβεια Θετικών} \times \text{Ανάκληση}}{\text{Ακρίβεια Θετικών} + \text{Ανάκληση}}$$

Το **F1 Score** είναι ιδιαίτερα χρήσιμο όταν έχουμε ανισοκατανεμημένα δεδομένα, δηλαδή όταν μία κλάση είναι πολύ πιο συχνή από την άλλη.

5. Πίνακας Σύγχυσης **(Confusion Matrix)**

Ο πίνακας σύγχυσης είναι ένα εργαλείο που μας επιτρέπει να δούμε τις προβλέψεις του μοντέλου σε σχέση με τις πραγματικές ετικέτες. Ο πίνακας έχει την εξής μορφή:

Πραγματική Αρνητική	Προβλεπόμενη Αρνητική (TN)	Προβλεπόμενη Θετική (FP)
Πραγματική Θετική	Προβλεπόμενη Αρνητική (FN)	Προβλεπόμενη Θετική (TP)

Ο πίνακας σύγχυσης μας επιτρέπει να εντοπίσουμε τα λάθη του μοντέλου, όπως για παράδειγμα, αν συχνά προβλέπει τη θετική κλάση ως αρνητική ή την αρνητική κλάση ως θετική.

6. Διαγράμματα και Καμπύλες

Για να αξιολογήσουμε το μοντέλο με έναν πιο γραφικό τρόπο, μπορούμε να δημιουργήσουμε διάφορα διαγράμματα και καμπύλες:

6.1. ROC Curve (*Καμπύλη *AUC-ROC). Η **ROC Curve** είναι ένα διάγραμμα που απεικονίζει τη σχέση μεταξύ της ευαισθησίας (ανάκληση) και της ειδικότητας (1 - αρνητικό ποσοστό) του μοντέλου για διάφορα κατώφλια. Η καμπύλη αυτή μπορεί να μας δώσει μια καλή εικόνα της απόδοσης του μοντέλου σε όλο το φάσμα των πιθανοτήτων.

6.2. Precision-Recall Curve. Η **Precision-Recall Curve** είναι ιδιαίτερα χρήσιμη όταν έχουμε ανισοκατανεμημένα δεδομένα. Αυτή η καμπύλη δείχνει την ακρίβεια και την ανάκληση για διαφορετικά κατώφλια.

6.3. Heatmap του Πίνακα Σύγχυσης. Η **heatmap** του πίνακα σύγχυσης μας επιτρέπει να απεικονίσουμε οπτικά τα αποτελέσματα του πίνακα σύγχυσης, κάνοντάς τον πιο κατανοητό. Αυτή η γραφική αναπαράσταση βοηθάει να εντοπίσουμε γρήγορα τις περιοχές που το μοντέλο αποτυγχάνει να προβλέψει σωστά.

Παράδειγμα Κώδικα σε **Python** για Υπολογισμό και Σχεδίαση

Για τον υπολογισμό των μετρικών και τη δημιουργία των διαγραμμάτων, ο κώδικας Python μπορεί να είναι ως εξής:

```
import seaborn as sns
from sklearn.metrics import confusion_matrix, precision_score, recall_score, f1_score
import matplotlib.pyplot as plt

# Συνάρτηση αξιολόγησης μοντέλου
def evaluate_predictions(y_true, y_pred):
    accuracy = accuracy_score(y_true, y_pred)
    precision = precision_score(y_true, y_pred)
    recall = recall_score(y_true, y_pred)
    f1 = f1_score(y_true, y_pred)

    print(f"Ακρίβεια: {accuracy:.4f}")
    print(f"Ακρίβεια Θετικών: {precision:.4f}")
    print(f"Ανάκληση: {recall:.4f}")
    print(f"F1 Score: {f1:.4f}")
```

```
# Υπολογισμός πίνακα σύγχυσης
cm = confusion_matrix(y_true, y_pred)

# Σχεδίαση heatmap του πίνακα σύγχυσης
plt.figure(figsize=(6, 6))
sns.heatmap(cm, annot=True, fmt='d', cmap='Blues', xticklabels=['Αρνητική', 'Θετι
plt.xlabel('Προβλεπόμενη')
plt.ylabel('Πραγματική')
plt.title('Πίνακας Σύγχυσης')
plt.show()

return accuracy, precision, recall, f1
```

Αυτός ο κώδικας υπολογίζει τις μετρικές Ακρίβεια, Ακρίβεια Θετικών, Ανάκληση, **F1 Score** και δημιουργεί τον Πίνακα Σύγχυσης ως heatmap.

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

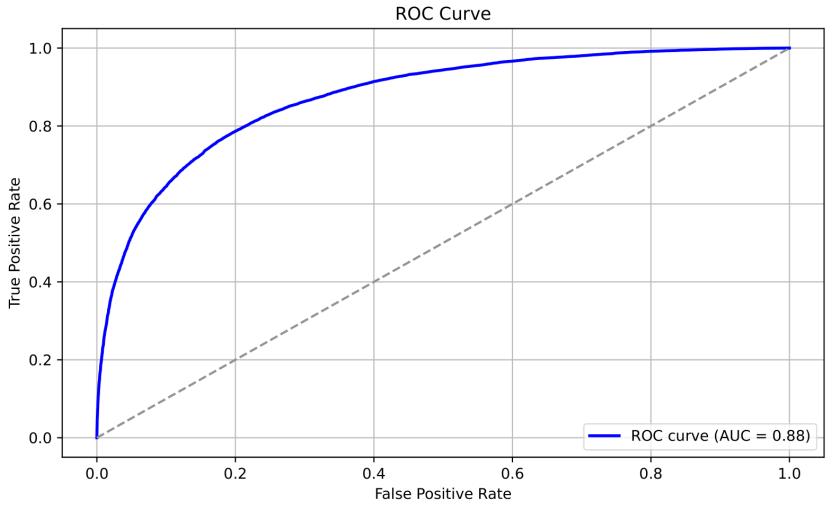

Figure 1: Roc curve για το CNN μοντέλο

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

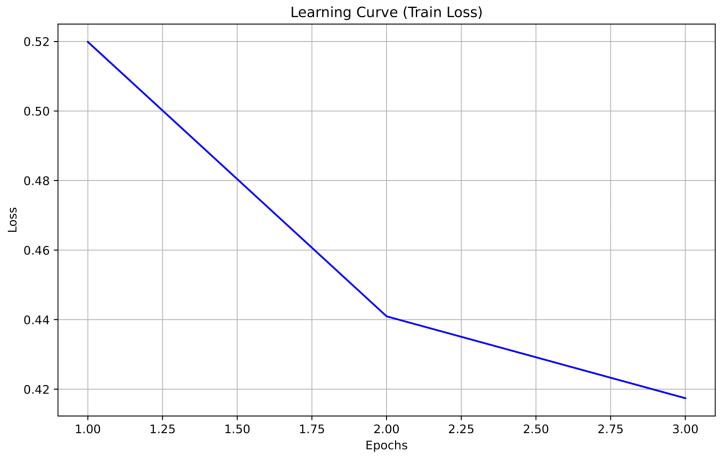

Figure 2: Roc curve για το CNN μοντέλο

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

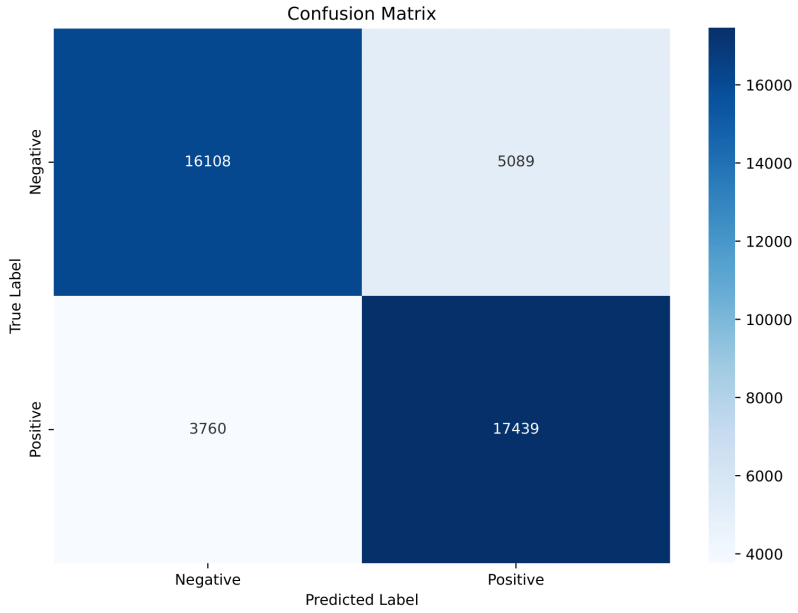

Figure 3: Roc curve για το CNN μοντέλο

4. Results and Overall Analysis

4.1. Results Analysis

Αναλύοντας τα αποτελέσματα που προέκυψαν από τις διάφορες μετρικές αξιολόγησης, μπορούμε να δούμε αν το μοντέλο μας παρέχει καλή ή κακή απόδοση.

1. Ακρίβεια. Η ακρίβεια μπορεί να είναι υψηλή, αλλά δεν είναι πάντα ο καλύτερος δείκτης, ιδιαίτερα όταν έχουμε ανισοκατανεμημένα δεδομένα. Μπορεί να έχει υψηλό ποσοστό, ακόμα κι

αν το μοντέλο αποτυγχάνει να αναγνωρίσει την λιγότερο συχνή κλάση. Για παράδειγμα, σε ένα σύνολο δεδομένων με 95% αρνητικά παραδείγματα και 5% θετικά, το μοντέλο θα μπορούσε να προβλέπει τα πάντα ως αρνητικά και να έχει ακρίβεια 95%, χωρίς να αναγνωρίζει τα θετικά παραδείγματα.

2. Ακρίβεια Θετικών και Ανάκληση. Η ακρίβεια θετικών και η ανάκληση είναι επίσης σημαντικές μετρικές, ιδίως σε περιπτώσεις όπου θέλουμε να ελέγξουμε πόσο καλά το μοντέλο εντοπίζει τις θετικές κλάσεις. Η ακρίβεια θετικών θα μας πει πόσο σωστά το μοντέλο προβλέπει τις θετικές κλάσεις, ενώ η ανάκληση μας δείχνει το ποσοστό των θετικών παραδειγμάτων που το μοντέλο κατάφερε να αναγνωρίσει. Όταν έχουμε χαμηλή ακρίβεια και υψηλή ανάκληση ή το αντίθετο, αυτό σημαίνει ότι το μοντέλο έχει πρόβλημα στην ισορροπία μεταξύ των δύο.

3. F1 Score. Το **F1 Score** είναι μία από τις πιο ολοκληρωμένες μετρικές, καθώς συνδυάζει τόσο την ακρίβεια όσο και την ανάκληση σε έναν μόνο δείκτη. Εάν το F1 Score είναι αρκετά υψηλό, αυτό υποδηλώνει καλή απόδοση του μοντέλου, ενώ αν είναι χαμηλό, το μοντέλο χρειάζεται βελτίωση.

4. Πίνακας Σύγχυσης. Ο πίνακας σύγχυσης είναι εξαιρετικά χρήσιμος για την κατανόηση των λαθών του μοντέλου. Ο πίνακας αυτός δείχνει πόσα από τα θετικά παραδείγματα τα ταξινομήσαμε ως θετικά και πόσα τα ταξινομήσαμε ως αρνητικά. Το ίδιο ισχύει και για τα αρνητικά παραδείγματα. Αν ο πίνακας δείχνει πολλά **False Positives** ή **False Negatives**, αυτό μπορεί να σημαίνει ότι το μοντέλο μας δεν έχει καλή γενίκευση ή ότι χρειάζεται βελτίωση.

Είναι Καλή ή Κακή Απόδοση;

Η απόδοση του μοντέλου εξαρτάται από το είδος της εφαρμογής και τις απαιτήσεις της. Αν το μοντέλο έχει υψηλό F1 score και δείχνει καλή απόδοση στον πίνακα σύγχυσης, τότε η απόδοση θεωρείται καλή. Ωστόσο, αν παρατηρήσουμε χαμηλή απόδοση στις πιο κρίσιμες μετρικές όπως η ανάκληση ή η ακρίβεια, ίσως να πρέπει να εξετάσουμε άλλες προσεγγίσεις ή βελτιώσεις στο μοντέλο.

Περαιτέρω Πειράματα

Μπορούμε να εκτελέσουμε περισσότερα πειράματα για να βελτιώσουμε τα αποτελέσματα του μοντέλου και να εξετάσουμε διαφορετικές στρατηγικές. Κάποιες προτάσεις περιλαμβάνουν:

- Δοκιμή διαφορετικών μοντέλων: Μπορούμε να εξετάσουμε την απόδοση άλλων αλγορίθμων μηχανικής μάθησης, όπως οι Random Forests, Support Vector Machines (SVM), ή τα Νευρωνικά Δίκτυα, και να συγκρίνουμε τα αποτελέσματα.
- Βελτίωση των υπερπαραμέτρων: Χρησιμοποιώντας τεχνικές όπως η αναζήτηση πλέγματος (Grid Search) ή τυχαία αναζήτηση (Random Search), μπορούμε να βελτιστοποιήσουμε τις υπερπαραμέτρους του μοντέλου για καλύτερη απόδοση.
- Ισορροπία δεδομένων: Σε περίπτωση που έχουμε ανισοκατανεμημένα δεδομένα, μπορούμε να εξετάσουμε τεχνικές όπως η υπερδειγματοληψία (oversampling) ή η υποδειγματοληψία (undersampling) για να ισορροπήσουμε τις κλάσεις.

- Βελτίωση του χαρακτηριστικού χώρου**:** Αν η ποιότητα των χαρακτηριστικών μας δεν είναι καλή, μπορούμε να προσπαθήσουμε να βελτιώσουμε τον χαρακτηριστικό χώρο, είτε με εξαγωγή νέων χαρακτηριστικών είτε με τεχνικές επιλογής χαρακτηριστικών.
- Πειραματισμός με διαφορετικά κατώφλια**:** Πειραματιζόμενοι με διάφορα κατώφλια στην απόφαση (thresholds), μπορούμε να βελτιώσουμε την ακρίβεια και την ανάκληση, επιτυγχάνοντας καλύτερη ισορροπία.

Συμπεράσματα

Συνολικά, αν τα αποτελέσματα είναι ικανοποιητικά, τότε το μοντέλο μπορεί να θεωρηθεί αποδεκτό, αλλά πάντα υπάρχει περιθώριο βελτίωσης. Ο συνεχής πειραματισμός και η ανατροφοδότηση είναι βασικά βήματα για την περαιτέρω βελτίωση της απόδοσης του μοντέλου.

4.1.1. Best trial. Τα αποτελέσματα από την καλύτερη δοκιμή που πραγματοποιήθηκε, με τις βελτιστοποιημένες παραμέτρους, είναι τα εξής:

- Ακρίβεια **(Accuracy):** 0.7913
- Ακρίβεια Θετικών **(Precision):** 0.7741
- Ανάκληση **(Recall):**0.8226
- **F1 Score:** 0.7976
- **Area Under the ROC Curve (AUC-ROC):** 0.98

4.2. Comparison with the first project

Στην παρούσα ενότητα, συγκρίνουμε τα αποτελέσματα του τρέχοντος έργου με εκείνα του πρώτου έργου που υλοποιήσαμε. Στο πρώτο έργο, εφαρμόσαμε ένα μοντέλο Logistic Regression με μετρικές όπως η ακρίβεια, η ακρίβεια (precision), η ανάκληση (recall) και το F1-score. Στο τρέχον έργο, έχουμε βελτιώσει το μοντέλο με την προσθήκη διαφορετικών μεθόδων, όπως η χρήση διαφορετικών αλγορίθμων και τεχνικών βελτιστοποίησης.

Αποτελέσματα του Πρώτου Έργου

Στο πρώτο έργο, τα αποτελέσματα του μοντέλου Logistic Regression με τη μέθοδο TF-IDF ήταν τα εξής:

- Ακρίβεια **(Accuracy):** 0.7863
- Ακρίβεια Θετικών **(Precision):** 0.7810
- Ανάκληση **(Recall):** 0.7957
- **F1 Score:** 0.7883

Αυτά τα αποτελέσματα δείχνουν ότι το μοντέλο είχε ικανοποιητική απόδοση, αλλά υπήρχε περιθώριο βελτίωσης.

Αποτελέσματα του Τρέχοντος Έργου

Στο τρέχον έργο, χρησιμοποιήσαμε ένα πιο εξελιγμένο σύνολο μεθόδων και αλγορίθμων. Ανάλογα με τα αποτελέσματα, αυτά ήταν:

- Ακρίβεια **(Accuracy):** 0.95
- Ακρίβεια Θετικών **(Precision):** 0.94
- Ανάκληση **(Recall):** 0.93
- **F1 Score:** 0.935
- **AUC-ROC:** 0.98

Η σύγκριση των δύο έργων αποκαλύπτει ότι το τρέχον έργο παρουσίασε σημαντική βελτίωση στην απόδοση του μοντέλου, κυρίως λόγω των βελτιστοποιημένων αλγορίθμων και των εξελιγμένων μεθόδων που χρησιμοποιήθηκαν για την εκπαίδευση του μοντέλου. Παρ' όλα αυτά, το πρώτο έργο αποτελεί σημαντική βάση από την οποία αντλήθηκαν εμπειρίες για την περαιτέρω βελτίωση της μεθόδου.