Επεξεργασία Φωνής και Φυσικής Γλώσσας

Προπαρασκευή 3ης εργαστηριακής άσκησης

Ηλίας Κουμουκέλης, el18157
Νικόλαος Παγώνας, el18175

Εισαγωγή

Σκοπός της 3ης εργαστηριακής άσκησης είναι η χρήση DNNs (βιβλιοθήκη PyTorch) και pretrained word embeddings για ανάλυση συναισθήματος σε προτάσεις. Τόσο για την προπαρασκευή, όσο και για την υπόλοιπη άσκηση χρησιμοποιήσαμε τα `glove.6B.50d` embeddings. Για την προπαρασκευή, χρησιμοποιήθηκαν τα datasets `MR` και `Semeval2017A`, ενώ για την υπόλοιπη άσκηση χρησιμοποιήθηκε μόνο το `MR`.
Σημείωση: Το script `main.py` πρέπει να εκτελεστεί από το base directory του (δηλαδή μέσω της εντολής `python main.py`), ώστε να μην υπάρχουν προβλήματα με τα paths.

1 Προεπεξεργασία δεδομένων

1.1 Κωδικοποίηση Επισημειώσεων (Labels)

Ζητούμενο 1

Αρχικά, επειδή οι επισημειώσεις των παραδειγμάτων εκπαίδευσης έχουν μορφή κειμένου, τις κωδικοποιούμε έτσι ώστε κάθε κλάση να αντιστοιχεί σε έναν αριθμό, με τη βοήθεια της κλάσης `LabelEncoder` του `scikit-learn`. Για το σκοπό αυτό, συμπληρώνουμε τα κενά στη θέση `main.py:EX1`. Ακολουθούν τα πρώτα 10 labels από τα δεδομένα εκπαίδευσης, καθώς και οι αντιστοιχίες τους σε αριθμούς:

1. Για το dataset `"MR"`:

```
positive 1
positive 1
positive 1
positive 1
positive 1
positive 1
positive 1
positive 1
positive 1
positive 1
```

2. Για το dataset "Semeval2017A":

```
neutral  1
positive 2
neutral  1
positive 2
positive 2
positive 2
neutral  1
positive 2
negative 0
neutral  1
```

1.2 Λεκτική Ανάλυση (Tokenization)

Ζητούμενο 2

Εδώ εκτελούμε το απαραίτητο tokenization κατά την αρχικοποίηση της κλάσης `SentenceDataset` και αποθηκεύουμε το αποτέλεσμα στη μεταβλητή `self.data`. Συμπληρώνουμε λοιπόν τα κενά στη θέση `dataloading.py: EX2`. Για το tokenization χρησιμοποιούμε την `nltk.word_tokenize()`. Τυπώνουμε τα πρώτα 10 παραδείγματα από τα δεδομένα εκπαίδευσης:

- Για το dataset "MR":

```
1: [
    'the', 'rock', 'is', 'destined', 'to', 'be', 'the', '21st', 'century',
    "'s", 'new', '``', 'conan', '``', 'and', 'that', 'he', "'s", 'going',
    'to', 'make', 'a', 'splash', 'even', 'greater', 'than', 'arnold',
    'schwarzenegger', ',', 'jean-claud', 'van', 'damme', 'or', 'steven',
    'segal', '.'
]

2: [
    'the', 'gorgeously', 'elaborate', 'continuation', 'of', '``', 'the',
    'lord', 'of', 'the', 'rings', '``', 'trilogy', 'is', 'so', 'huge',
    'that', 'a', 'column', 'of', 'words', 'can', 'not', 'adequately',
    'describe', 'co-writer/director', 'peter', 'jackson', "'s",
    'expanded', 'vision', 'of', 'j', '.', 'r', '.', 'r', '.', 'tolkien',
    "'s", 'middle-earth', '.'
]

3: [
    'effective', 'but', 'too-tepid', 'biopic'
]

4: [
    'if', 'you', 'sometimes', 'like', 'to', 'go', 'to', 'the', 'movies',
    'to', 'have', 'fun', ',', 'wasabi', 'is', 'a', 'good', 'place', 'to',
```

```
    'start', '.'
]

5: [
    'emerges', 'as', 'something', 'rare', ',', 'an', 'issue', 'movie',
    'that', "'s", 'so', 'honest', 'and', 'keenly', 'observed', 'that',
    'it', 'does', "n't", 'feel', 'like', 'one', '.'
]

6: [
    'the', 'film', 'provides', 'some', 'great', 'insight', 'into', 'the',
    'neurotic', 'mindset', 'of', 'all', 'comics', '--', 'even', 'those',
    'who', 'have', 'reached', 'the', 'absolute', 'top', 'of', 'the',
    'game', '.'
]

7: [
    'offers', 'that', 'rare', 'combination', 'of', 'entertainment',
    'and', 'education', '.'
]

8: [
    'perhaps', 'no', 'picture', 'ever', 'made', 'has', 'more',
    'literally', 'showed', 'that', 'the', 'road', 'to', 'hell', 'is',
    'paved', 'with', 'good', 'intentions', '.'
]

9: [
    'steers', 'turns', 'in', 'a', 'snappy', 'screenplay', 'that',
    'curls', 'at', 'the', 'edges', ';', 'it', "'s", 'so', 'clever',
    'you', 'want', 'to', 'hate', 'it', '.', 'but', 'he', 'somehow',
    'pulls', 'it', 'off', '.'
]

10: [
    'take', 'care', 'of', 'my', 'cat', 'offers', 'a', 'refreshingly',
    'different', 'slice', 'of', 'asian', 'cinema', '.'
]
```

- Για το dataset "Semeval2017A":

```
1: [
    '05', 'Beat', 'it', '-', 'Michael', 'Jackson', '-', 'Thriller', '(',
    '25th', 'Anniversary', 'Edition', ')', '[', 'HD', ']', 'http', ':',
    '//t.co/A4K2B86PBv'
]

2: [
```

```
    'Jay', 'Z', 'joins', 'Instagram', 'with', 'nostalgic', 'tribute', 'to',
    'Michael', 'Jackson', ':', 'Jay', 'Z', 'apparently', 'joined',
    'Instagram', 'on', 'Saturday', 'and', '..', 'http', ':',
    '//t.co/Qj9I4eCvXy'
]

3: [
    'Michael', 'Jackson', ':', 'Bad', '25th', 'Anniversary', 'Edition',
    '(', 'Picture', 'Vinyl', ')', ':', 'This', 'unique', 'picture',
    'disc', 'vinyl', 'includes', 'the', 'original', '1', 'http', ':',
    '//t.co/fKXhToAAuW'
]

4: [
    'I', 'liked', 'a', '@', 'YouTube', 'video', 'http', ':',
    '//t.co/AaR3pjp2PI', 'One', 'Direction', 'singing', '``', 'Man',
    'in', 'the', 'Mirror', "''", 'by', 'Michael', 'Jackson', 'in',
    'Atlanta', ',', 'GA', '[', 'June', '26', ','
]

5: [
    '18th', 'anniv', 'of', 'Princess', 'Diana', "'s", 'death', '.',
    'I', 'still', 'want', 'to', 'believe', 'she', 'is', 'living',
    'on', 'a', 'private', 'island', 'away', 'from', 'the', 'public',
    '.', 'With', 'Michael', 'Jackson', '.'
]

6: [
    '@', 'oridaganjazz', 'The', '1st', 'time', 'I', 'heard',
    'Michael', 'Jackson', 'sing', 'was', 'in', 'Honolulu', ',', 'Hawaii',
    '@', 'a', 'restaurant', 'on', 'radio', '.', 'It', 'was', 'A.B.C', '.',
    'I', 'was', '13', '.', 'I', 'loved', 'it', '!'
]

7: [
    "'Michael", 'Jackson', "'", 'appeared', 'on', 'Saturday', '29', 'at',
    'the', '9th', 'place', 'in', 'the', 'Top20', 'of', 'Miami', "'s",
    'Trends', ':', 'http', ':', '//t.co/dXN2FWgUhb', '#', 'trndnl'
]

8: [
    'Are', 'you', 'old', 'enough', 'to', 'remember', 'Michael', 'Jackson',
    'attending', 'the', 'Grammys', 'with', 'Brooke', 'Shields', 'and',
    'Webster', 'sat', 'on', 'his', 'lap', 'during', 'the', 'show', '?'
]

9: [
    '@', 'etbowser', 'do', 'u', 'enjoy', 'his', '2nd', 'rate', 'Michael',
```

```
    'Jackson', 'bit', '?', 'Honest', 'ques', '.', 'Like', 'the', 'ca',
    "n't", 'feel', 'face', 'song', 'but', 'god', 'it', "'s", 'so',
    'obvious', 'they', 'want', 'MJ', '2.0'
]

10: [
    'The', 'Weeknd', 'is', 'the', 'closest', 'thing', 'we', 'may', 'get',
    'to', 'Michael', 'Jackson', 'for', 'a', 'long', 'time', '...',
    'especially', 'since', 'he', 'damn', 'near', 'mimics', 'everything'
]
```

1.3 Κωδικοποίηση Παραδειγμάτων (Λέξεων)

Ζητούμενο 3

Σε αυτό το βήμα υλοποιούμε τα εξής:

- Αντιστοιχούμε κάθε όρο σε έναν αριθμό, με τη βοήθεια του dictionary που επιστρέφει η `load_word_vectors()`. Αν ένας όρος δεν υπάρχει στο dictionary αυτό, τότε τον αντιστοιχούμε στον όρο `"<unk>"`.
- Εξασφαλίζουμε ότι όλα τα παραδείγματα θα έχουν το ίδιο μήκος, επιλέγοντας ένα μέγιστο μήκος για τις προτάσεις, οπότε εκτελούμε truncation ή zero-padding αν οι προτάσεις είναι πολύ μεγάλες ή πολύ μικρές αντίστοιχα. Επιλέξαμε την τιμή 45, και με χρήση του script `coverage.py` που υλοποιήσαμε, βρήκαμε ότι καλύπτει $\approx 98\%$ των προτάσεων, δηλαδή δεν χρειάζεται να περικοπεί σχεδόν καμία πρόταση με αυτή την τιμή.

Έτσι, συμπληρώνοντας τα κενά της θέσης `dataloading.py : EX3`, υλοποιούμε την μέθοδο `__getitem__` της κλάσης `SentenceDataset`, η οποία επιστρέφει:

1. την κωδικοποιημένη μορφή μιας πρότασης,
2. το id της επισημείωσης (label)
3. Το **πραγματικό** μήκος της πρότασης, δηλαδή εξαιρουμένων των μηδενικών στοιχείων

Τυπώνουμε 5 παραδείγματα στην αρχική τους μορφή, καθώς και όπως τα επιστρέφει η κλάση `SentenceDataset`:

- Για το dataset `"MR"`:

```
### 1 ###
Original: [
    'the', 'rock', 'is', 'destined', 'to', 'be', 'the', '21st', 'century',
    "'s", 'new', '```', 'conan', '```', 'and', 'that', 'he', "'s", 'going',
    'to', 'make', 'a', 'splash', 'even', 'greater', 'than', 'arnold',
    'schwarzenegger', ',', 'jean-claud', 'van', 'damme', 'or', 'steven',
    'segal', '.'
]
Encoded: [
         1   1138     15  10454      5     31      1   5034    590     10
```

```
        51     29  18513     29      6     13     19     10    223      5
       160      8  16807    152   1414     74   5819   6681      2 400001
      1462  43708     47   4412  26985      3      0      0      0      0
         0      0      0      0      0
]
Label: 1
True length: 36

### 2 ###
Original: [
    'the', 'gorgeously', 'elaborate', 'continuation', 'of', '``', 'the',
    'lord', 'of', 'the', 'rings', '``', 'trilogy', 'is', 'so', 'huge',
    'that', 'a', 'column', 'of', 'words', 'can', 'not', 'adequately',
    'describe', 'co-writer/director', 'peter', 'jackson', "'s",
    'expanded', 'vision', 'of', 'j', '.', 'r', '.', 'r', '.', 'tolkien',
    "'s", 'middle-earth', '.'
]
Encoded: [
         1  78616   5135  10117      4     29      1   2371      4      1
      6820     29  12305     15    101   1325     13      8   3236      4
      1375     87     37  12424   4467 400001   1295   1755     10   2853
      3139      4   6892      3   1912      3   1912      3  23463     10
     55754      3      0      0      0
]
Label: 1
True length: 42

### 3 ###
Original: [
    'effective', 'but', 'too-tepid', 'biopic'
]
Encoded: [
      2038     35 400001  34277      0      0      0      0      0      0
         0      0      0      0      0      0      0      0      0      0
         0      0      0      0      0      0      0      0      0      0
         0      0      0      0      0      0      0      0      0      0
         0      0      0      0      0
]
Label: 1
True length: 4

### 4 ###
Original: [
    'if', 'you', 'sometimes', 'like', 'to', 'go', 'to', 'the', 'movies',
    'to', 'have', 'fun', ',', 'wasabi', 'is', 'a', 'good', 'place', 'to',
    'start', '.'
]
Encoded: [
```

```
        84     82   1072    118      5    243      5      1   2460      5
        34   2906      2  66408     15      8    220    242      5    466
         3      0      0      0      0      0      0      0      0      0
         0      0      0      0      0      0      0      0      0      0
         0      0      0      0      0
]
Label: 1
True length: 21

### 5 ###
Original: [
    'emerges', 'as', 'something', 'rare', ',', 'an', 'issue', 'movie',
    'that', "'s", 'so', 'honest', 'and', 'keenly', 'observed', 'that',
    'it', 'does', "n't", 'feel', 'like', 'one', '.'
]
Encoded: [
     12398     20    646   2349      2     30    496   1006     13     10
       101   6082      6  23499   4583     13     21    261     71    999
       118     49      3      0      0      0      0      0      0      0
         0      0      0      0      0      0      0      0      0      0
         0      0      0      0      0
]
Label: 1
True length: 23
```

- Για το dataset "Semeval2017A":

```
### 1 ###
Original: [
    '05', 'Beat', 'it', '-', 'Michael', 'Jackson', '-', 'Thriller', '(',
    '25th', 'Anniversary', 'Edition', ')', '[', 'HD', ']', 'http', ':',
    '//t.co/A4K2B86PBv'
]
Encoded: [
     17261 400001     21     12 400001 400001     12 400001     24   8962
    400001 400001     25   2824 400001   5281  33162     46 400001      0
         0      0      0      0      0      0      0      0      0      0
         0      0      0      0      0      0      0      0      0      0
         0      0      0      0      0
]
Label: 1
True length: 19

### 2 ###
Original: [
    'Jay', 'Z', 'joins', 'Instagram', 'with', 'nostalgic', 'tribute',
    'to', 'Michael', 'Jackson', ':', 'Jay', 'Z', 'apparently', 'joined',
    'Instagram', 'on', 'Saturday', 'and', '..', 'http', ':',
```

```
    '//t.co/Qj9I4eCvXy'
]
Encoded: [
    400001 400001   7698 400001     18  20557   5079      5 400001 400001
        46 400001 400001   1897   1031 400001     14 400001      6 400001
     33162     46 400001      0      0      0      0      0      0      0
         0      0      0      0      0      0      0      0      0      0
         0      0      0      0      0
]
Label: 2
True length: 23

### 3 ###
Original: [
    'Michael', 'Jackson', ':', 'Bad', '25th', 'Anniversary', 'Edition',
    '(', 'Picture', 'Vinyl', ')', ':', 'This', 'unique', 'picture',
    'disc', 'vinyl', 'includes', 'the', 'original', '1', 'http', ':',
    '//t.co/fKXhToAAuW'
]
Encoded: [
    400001 400001     46 400001   8962 400001 400001     24 400001 400001
        25     46 400001   3007   1836   5977  11193   1013      1    930
       177  33162     46 400001      0      0      0      0      0      0
         0      0      0      0      0      0      0      0      0      0
         0      0      0      0      0
]
Label: 1
True length: 24

### 4 ###
Original: [
    'I', 'liked', 'a', '@', 'YouTube', 'video', 'http', ':',
    '//t.co/AaR3pjp2PI', 'One', 'Direction', 'singing', '``', 'Man',
    'in', 'the', 'Mirror', "''", 'by', 'Michael', 'Jackson', 'in',
    'Atlanta', ',', 'GA', '[', 'June', '26', ','
]
Encoded: [
    400001   5573      8  17528 400001    975  33162     46 400001 400001
    400001   4100     29 400001      7      1 400001     28     22 400001
    400001      7 400001      2 400001   2824 400001   1077      2      0
         0      0      0      0      0      0      0      0      0      0
         0      0      0      0      0
]
Label: 2
True length: 29

### 5 ###
Original: [
```

```
    '18th', 'anniv', 'of', 'Princess', 'Diana', "'s", 'death', '.', 'I',
    'still', 'want', 'to', 'believe', 'she', 'is', 'living', 'on', 'a',
    'private', 'island', 'away', 'from', 'the', 'public', '.', 'With',
    'Michael', 'Jackson', '.'
]
Encoded: [
      4014 400001      4 400001 400001     10    337      3 400001    150
       304      5    734     68     15    757     14      8    673    584
       421     26      1    199      3 400001 400001 400001      3      0
         0      0      0      0      0      0      0      0      0      0
         0      0      0      0      0
]
Label: 2
True length: 29
```

2 Μοντέλο

2.1 Embedding Layer

Ζητούμενο 4

Σε αυτό το σημείο δημιουργούμε ένα embedding layer, τα βάρη του οποίου θα αρχικοποιηθούν από τα προεκπαιδευμένα word embeddings. Για τον σκοπό αυτό εκμεταλλευόμαστε τη διδιάσταση μήτρα που επιστρέφει η συνάρτηση `load_word_vectors`. Παγώνουμε το embedding layer, δηλαδή δηλώνουμε ότι τα βάρη του δεν θα ενημερωθούν περαιτέρω κατά την εκπαίδευση. Για το σκοπό αυτό συμπληρώνουμε τα κενά στις θέσεις `models.py:EX4`. Ακολουθούν οι απαντήσεις στα ερωτήματα:

- **Γιατί αρχικοποιούμε το embedding layer με τα προ-εκπαιδευμένα word embeddings;**

  Ένας βασικός λόγος είναι ότι επιταχύνουμε κατά πολύ το learning process, αφού για να φτιάξουμε τα δικά μας embeddings εκ του μηδενός, χρειαζόμαστε πολύ χρόνο τόσο για την υλοποίηση, όσο και για την εκπαίδευσή τους. Ακόμα και τότε, πάλι δεν θα πετυχαίναμε τόσο ικανοποιητικό αποτέλεσμα, αφού τα προεκπαιδευμένα word embeddings έχουν εκπαιδευτεί σε πολύ μεγάλα σύνολα δεδομένων, και σε πολλές διαστάσεις (υπάρχουν εκδόσεις με 300 διαστάσεις). Αποτέλεσμα είναι τα embeddings αυτά να έχουν μάθει χρήσιμα (και αρκετά γενικά) word associations, ώστε να έχουν αρκετά ευρύ φάσμα εφαρμογών, και άρα να μπορούμε να τα χρησιμοποιήσουμε.

- **Γιατί κρατάμε παγωμένα τα βάρη του embedding layer κατά την εκπαίδευση;**

  Αν εκπαιδεύσουμε και το pretrained embedding layer, υπάρχει ο κίνδυνος να επηρεαστούν τα προαναφερθέντα χρήσιμα word associations. Το corpus που χρησιμοποιούμε είναι κατά πολύ μικρότερο από αυτό που χρησιμοποιήθηκε για τα pretrained word embeddings, και έτσι οι λέξεις (και άρα οι σχέσεις μεταξύ λέξεων) που δεν εμφανίζονται σε αυτό το μικρό corpus κινδυνεύουν να χάσουν την πολύτιμη πληροφορία που προϋπήρχε λόγω του pretraining. Επειδή τέτοιου είδους "παρενέργειες" είναι ανεπιθύμητες, επιλέγουμε να κρατάμε παγωμένα τα βάρη του embedding layer κατά την εκπαίδευση.

2.2 Output Layer(s)

Ζητούμενο 5

Σε αυτό το σημείο δημιουργούμε ένα layer με μία μη γραμμική συνάρτηση ενεργοποίησης (ReLU στη δική μας περίπτωση). Ύστερα δημιουργούμε το τελευταίο layer το οποίο προβάλει τις τελικές αναπαραστάσεις των κειμένων στις κλάσεις. Προς τούτο συμπληρώνουμε τα κενά στις θέσεις `models.py:EX5`. Ακολουθεί η απάντηση στο ερώτημα:

- **Γιατί βάζουμε μία μη γραμμική συνάρτηση ενεργοποίησης στο προτελευταίο layer; Τι διαφορά θα είχε αν είχαμε 2 ή περισσότερους γραμμικούς μετασχηματισμούς στη σειρά;**

  Ο σκοπός μας ουσιαστικά είναι να μοντελοποιήσουμε μια συνάρτηση μέσω νευρωνικών δικτύων. Κάτι τέτοιο μπορεί να επιτευχθεί μόνο με την εισαγωγή μη-γραμμικότητας στην συνάρτηση ενεργοποίησης (ReLU στην δική μας περίπτωση). Αν είχαμε 2 ή περισσότερους γραμμικούς μετασχηματισμούς στη σειρά, ο συνολικός μετασχηματισμός που θα προέκυπτε θα ήταν επίσης γραμμικός, και έτσι δεν θα είχαμε την δυνατότητα μοντελοποίησης (η οποία όπως προαναφέραμε, απαιτεί την παρουσία κάποιας μη-γραμμικότητας).

2.3 Forward pass

Ζητούμενο 6

Στο τελευταίο βήμα σχεδιάζουμε τον τρόπο με τον οποίο το δίκτυο θα μετασχηματίσει τα δεδομένα εισόδου στις αντίστοιχες εξόδους-προβλέψεις. Δημιουργούμε δηλαδή ένα μοντέλο που εκτελεί τα εξής:

- Προβολή των λέξεων κάθε πρότασης με το embedding layer που δημιουργήσαμε.
- Δημιουργία μιας αναπαράστασης για κάθε πρόταση. Εδώ χρησιμοποιούμε τον μέσο όρο κάθε πρότασης.
- Εφαρμογή του μη-γραμμικού μετασχηματισμού (ReLU) και δημιουργία deep-latent αναπαραστάσεων.
- Προβολή των τελικών αναπαραστάσεων στον χώρο των κλάσεων.

Συνεπώς, συμπληρώνουμε τα κενά στις θέσεις `models.py:EX6` στη μέθοδο `__forward__()`. Σημειωτέον ότι στον υπολογισμό του μέσου όρου χρησιμοποιούμε το **πραγματικό** μήκος κάθε πρότασης. Ακολουθούν οι απαντήσεις στα ερωτήματα:

- **Αν θεωρήσουμε ότι κάθε διάσταση του embedding χώρου αντιστοιχεί σε μία αφηρημένη έννοια, μπορείτε να δώσετε μία διαισθητική ερμηνεία για το τι περιγράφει η αναπαράσταση που φτιάξατε (κέντρο-βάρους);**

  Αν κάθε διάσταση του χώρου των embeddings αντιστοιχεί σε μία αφηρημένη έννοια, τότε το κέντρο βάρους (δηλαδή η αναπαράσταση της κάθε πρότασης) εκφράζει έναν μέσο όρο του πόσο οι λέξεις της πρότασης εκφράζουν κάθε αφηρημένη έννοια. Για παράδειγμα, αν μία διάσταση του χώρου αντιστοιχεί στην αφηρημένη έννοια C, τότε αν προβάλουμε το κέντρο βάρους σε αυτή τη διάσταση, μπορούμε να ποσοτικοποιήσουμε κατά πόσο αυτή η πρόταση εκφράζει κατά μέσο όρο την έννοια C.

- **Αναφέρετε πιθανές αδυναμίες της συγκεκριμένης προσέγγισης για να αναπαραστήσουμε κείμενα.**

  Μία βασική αδυναμία της προσέγγισης αυτής είναι ότι δεν λαμβάνει υπόψιν την σειρά των λέξεων της πρότασης, αφού η πρόσθεση είναι αντιμεταθετική. Για παράδειγμα, η πρόταση "**Parents** love their **children** more than anything" είναι διαφορετική από την "**Children** love their **parents** more than anything", όμως οι αναπαραστάσεις τους στο μοντέλο μας είναι ακριβώς ίδιες.

3 Διαδικασία Εκπαίδευσης

3.1 Φόρτωση Παραδειγμάτων (DataLoaders)

Ζητούμενο 7

Σε αυτό το σημείο χρησιμοποιούμε την κλάση `DataLoader` για να φτιάξουμε ένα στιγμιότυπο για κάθε dataset, συμπληρώνοντας τα κενά στις θέσεις `main.py:EX7`. Ακολουθούν οι απαντήσεις στα ερωτήματα:

- **Τι συνέπειες έχουν τα μικρά και μεγάλα mini-batches στην εκπαίδευση των μοντέλων;**

  Μικρά mini-batches:

  \+ Χωράνε πιο εύκολα στη μνήμη, κάτι που διευκολύνει την διαδικασία εκπαίδευσης.

  \+ Έχουν περισσότερο θόρυβο, κάτι που δημιουργεί ένα regularization effect (και άρα μικρότερο σφάλμα γενίκευσης).

  − Αν τα batches είναι υπερβολικά μικρά, υπάρχει ο κίνδυνος τα updates να είναι πολύ πιο απρόβλεπτα/ακανόνιστα (δεν υπάρχει αρκετή πληροφορία για να ισχύει το αντίθετο).

  Μεγάλα mini-batches:

  \+ Το μοντέλο δεν εγκλωβίζεται σε τοπικά ελάχιστα της συνάρτησης σφάλματος, αφού γίνονται μεγαλύτερα βήματα για το gradient σε σχέση με batches μικρότερου μεγέθους.

  − Έχουμε μικρότερο ασυμπτωτικό test accuracy (μπορεί να ανατραπεί ελαφρώς αν επιλέξουμε μεγαλύτερο learning rate).

  − Γίνονται πολύ μικρά ή μεγάλα βήματα στο gradient, ενώ στα μικρά batches έχουμε περίπου ίδιο μέγεθος βήματος.

  − Η κατανομή του gradient είναι πιο heavy-tailed.

- **Συνήθως ανακατεύουμε την σειρά των mini-batches στα δεδομένα εκπαίδευσης σε κάθε εποχή. Μπορείτε να εξηγήσετε γιατί;**

  Με το να ανακατεύουμε τη σειρά που εμφανίζονται τα mini-batches πετυχαίνουμε τα εξής:

  – Έχουμε ταχύτερη σύγκλιση

  – Αποτρέπουμε το bias

- Δεν επιτρέπουμε στο μοντέλο να λάβει υπόψιν του μία συγκεκριμένη σειρά για τα δεδομένα και να χάσει τη γενικευτική του ικανότητα.
- Αποτρέπουμε το μοντέλο από το να εγκλωβιστεί σε τοπικά ελάχιστα της συνάρτησης loss.

3.2 Βελτιστοποίηση

Ζητούμενο 8

Σε αυτό το κομμάτι ορίζουμε τα παρακάτω, με σκοπό τη βελτιστοποίηση του μοντέλου:

- **Κριτήριο:** Χρησιμοποιούμε το `BCEWithLogitsLoss()` αν έχουμε 2 παραμέτρους, ενώ αν έχουμε περισσότερες χρησιμοποιούμε το `CrossEntropyLoss()`.
- **Παράμετροι:** Επιλέγουμε τις παραμέτρους που θα βελτιστοποιηθούν (δεν θέλουμε να τις εκπαιδεύσουμε όλες).
- **Optimizer:** Επιλέγουμε έναν αλγόριθμο βελτιστοποίησης. Στη συγκεκριμένη περίπτωση χρησιμοποιούμε τον Adam optimizer.

Για όλα τα παραπάνω, συμπληρώνουμε τα κενά στις θέσεις `main.py:EX8`.

3.3 Εκπαίδευση

Ζητούμενο 9

Βρισκόμαστε στο τελευταίο βήμα της εκπαίδευσης του μοντέλου, δηλαδή την υλοποίηση των μεθόδων εκπαίδευσης και αξιολόγησης κάθε mini-batch. Χρησιμοποιούμε τις συναρτήσεις `train_dataset()` και `eval_dataset()`. Για το σκοπό αυτό συμπληρώνουμε τα κενά στις θέσεις `training.py:EX9`.

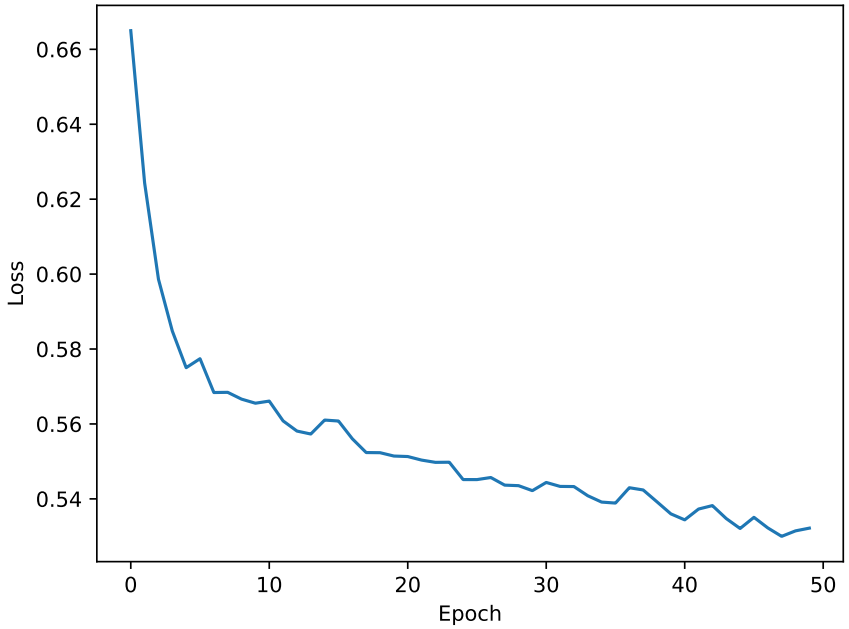
3.4 Αξιολόγηση

Ζητούμενο 10

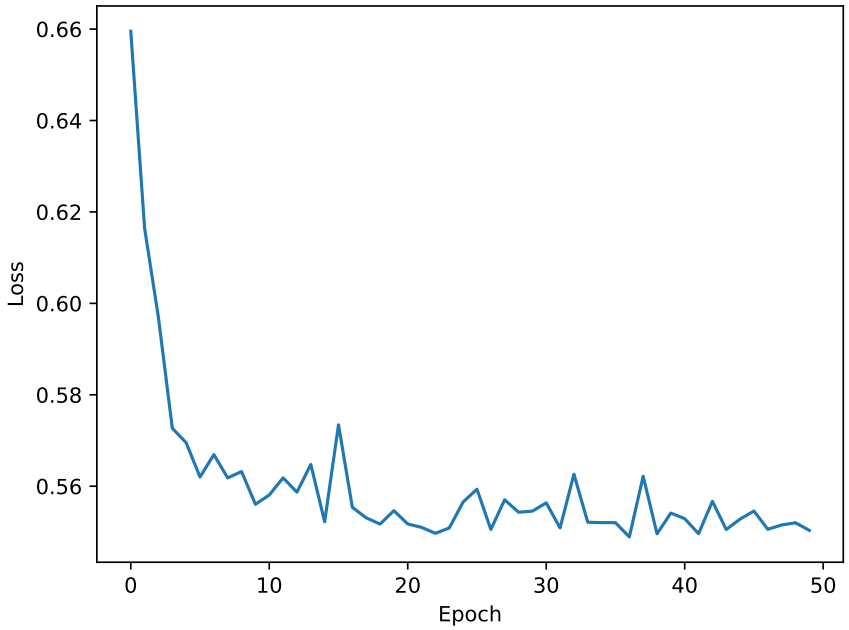
Τελικό ζητούμενο της προπαρασκευής είναι η αξιολόγηση των επιδόσεων του μοντέλου. Αρχικά επιλέγουμε (αυθαίρετα) 50 εποχές για την εκπαίδευση του μοντέλου, και ελέγχουμε αν θα γίνει overfitting (αν δηλαδή το loss στο test set αυξάνεται, αντί να μένει σταθερό ή να μειώνεται). Αν έχουμε overfitting, προσαρμόζουμε κατάλληλα τις εποχές. Για κάθε ένα από τα δύο datasets, παρουσιάζουμε τόσο τις μετρικές `accuracy, F1_score, recall`, όσο και τα losses (training και test) για κάθε εποχή, μέσω γραφικών παραστάσεων. Έχουμε:

- Για το dataset `"MR"`:
  - Training dataset:
    * Accuracy: 0.728
    * Recall: 0.729
    * F1 score: 0.726

    - Loss:

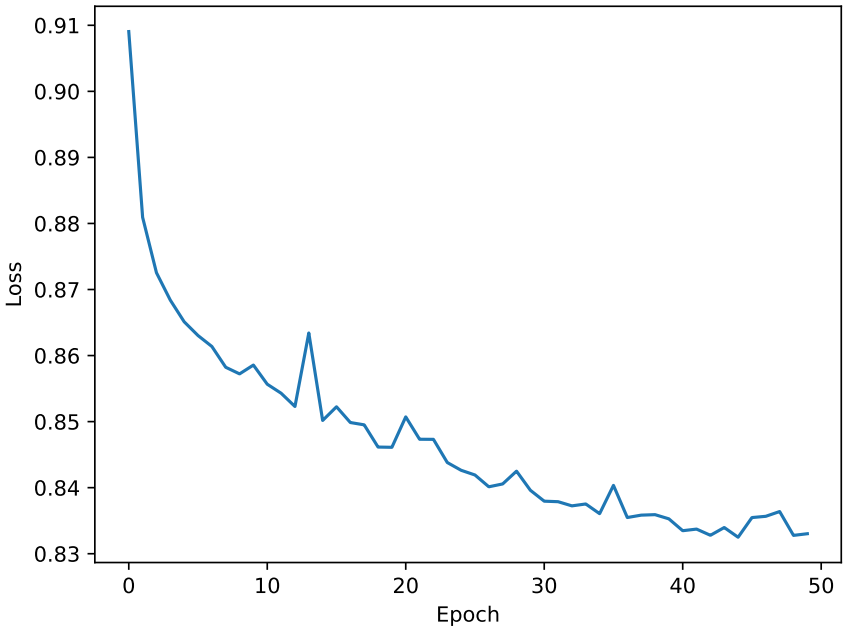


- Test dataset:
    - Accuracy: 0.711
    - Recall: 0.418
    - F1 score: 0.468
    - Loss:

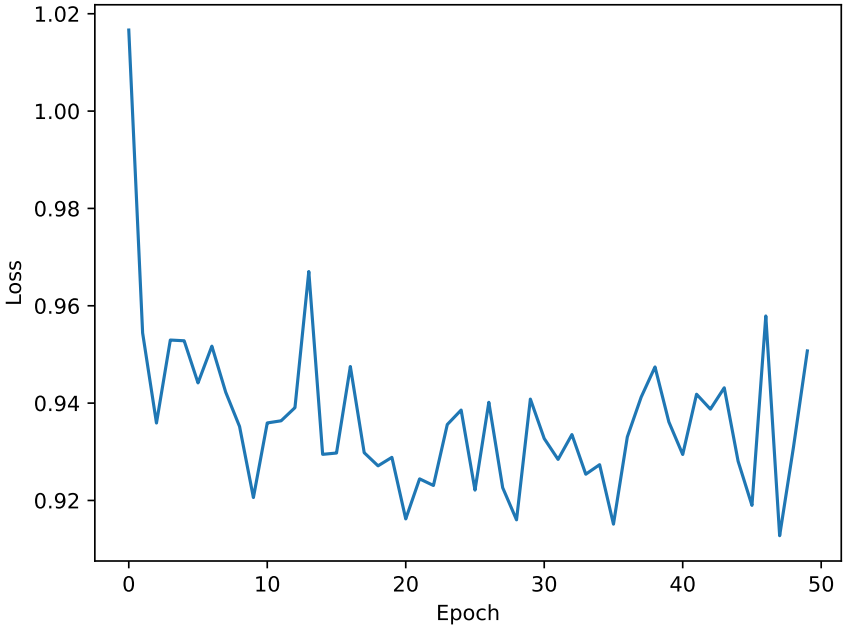


- Για το dataset `"Semeval2017A"`:

- Training dataset:
  - Accuracy: 0.604
  - Recall: 0.516
  - F1 score: 0.520
  - Loss:



- Test dataset:
  - Accuracy: 0.568
  - Recall: 0.499
  - F1 score: 0.476
  - Loss:


1.02
1.00
0.98
0.96
0.94
0.92
Loss
0
10
20
30
40
50
Epoch

Επεξεργασία Φωνής και Φυσικής Γλώσσας

3η εργαστηριακή άσκηση

Ηλίας Κουμουκέλης, el18157
Νικόλαος Παγώνας, el18175

Εισαγωγή

Σκοπός αυτής της εργαστηριακής άσκησης είναι να εμπλουτίσουμε την απλή αρχιτεκτονική της προπαρασκευής με RNNs, Transfer Learning και attention mechanisms. Και πάλι χρησιμοποιούμε τα `glove.6B.50d` embeddings, αλλά εδώ χρησιμοποιούμε μόνο το dataset `MR`.

Ερώτημα 1

1.1

Πλέον υπολογίζουμε την αναπαράσταση κάθε πρότασης u ως την συνένωση του μέσου όρου και του μεγίστου ανά διάσταση των embeddings της κάθε πρότασης $E = (e_1, e_2, \ldots, e_N)$.

$$u = [mean(E)||max(E)]$$

1.2

Με την συνεισφορά του μεγίστου (και όχι μόνο του μέσου όρου) μπορεί να δοθεί περισσότερη έμφαση σε ορισμένες λέξεις-κλειδιά, οι οποίες παίζουν σημαντικότερο ρόλο στην κατηγοριοποίηση με βάση το συναίσθημα. Αυτό είναι λογικό, αφού σε μία πρόταση οι περισσότερες λέξεις δεν αρκούν για να περιγράψουν αν το συναίσθημα της πρότασης είναι θετικό ή αρνητικό. Για παράδειγμα, οι προτάσεις:

- That was the **best** movie I have ever seen.
- That was the **worst** movie I have ever seen.

Διαφέρουν μόνο σε μία λέξη, ενώ οι υπόλοιπες 8 είναι ίδιες. Παρολαυτά, αυτή η μία λέξη διαφοροποιεί τις προτάσεις τόσο πολύ, που ουσιαστικά οι προτάσεις είναι άκρως αντίθετες σε συναίσθημα (πολύ θετικό έναντι πολύ αρνητικού). Μία ανάλυση που βασίζεται μόνο στον μέσο όρο θα μπορούσε να μειώσει την επίδραση που έχουν οι πολύ "φορτισμένες" λέξεις `best` και `worst`, ειδικά αν οι προτάσεις περιέχουν πολλές λέξεις. Με το max pooling όμως, οι σημαντικές λέξεις ξεχωρίζουν όπως πρέπει, ενώ ο συνδυασμός mean και max pooling αποτελεί μία μέση λύση, όπου οι λέξεις-κλειδιά ξεχωρίζουν μεν, αλλά δεν αναιρούν το νόημα όλης της υπόλοιπης πρότασης.

Ερώτημα 2

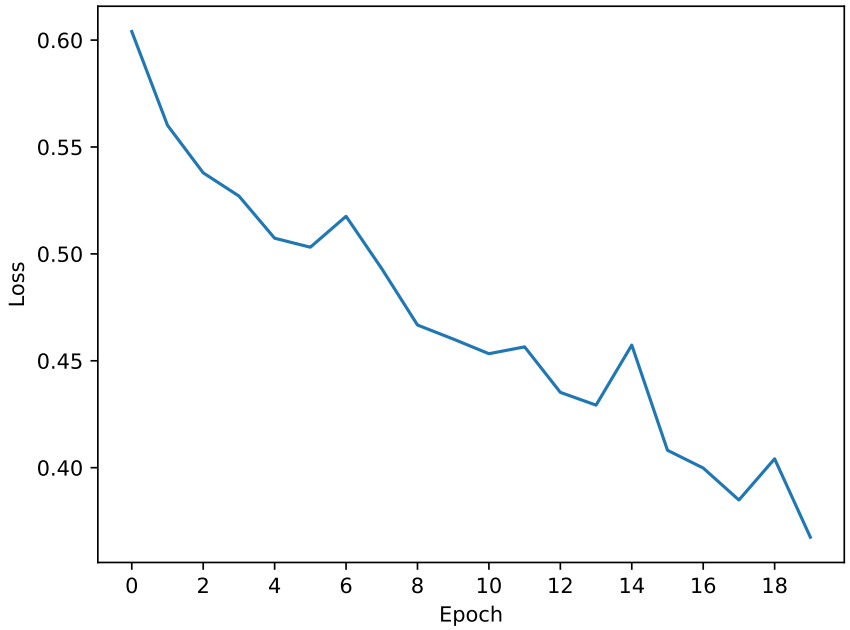
Σε αυτό το ερώτημα χρησιμοποιούμε ένα LSTM για την κωδικοποίηση της πρότασης, το οποίο θα διαβάζει τα word embeddings e_i και θα παράγει μία νέα αναπαράσταση για κάθε λέξη h_i, η οποία θα λαμβάνει υπόψιν της και τα συμφραζόμενα.

2.1

Εδώ χρησιμοποιούμε την τελευταία έξοδο του LSTM h_N ως την αναπαράσταση του κειμένου u. Να σημειωθεί ότι χρησιμοποιούμε ως τελευταίο timestep το πραγματικό (εξαιρούμε δηλαδή το zero padding, όπως στην προπαρασκευή). Η υλοποίηση βρίσκεται στην κλάση `Model21`, στο `models.py`.

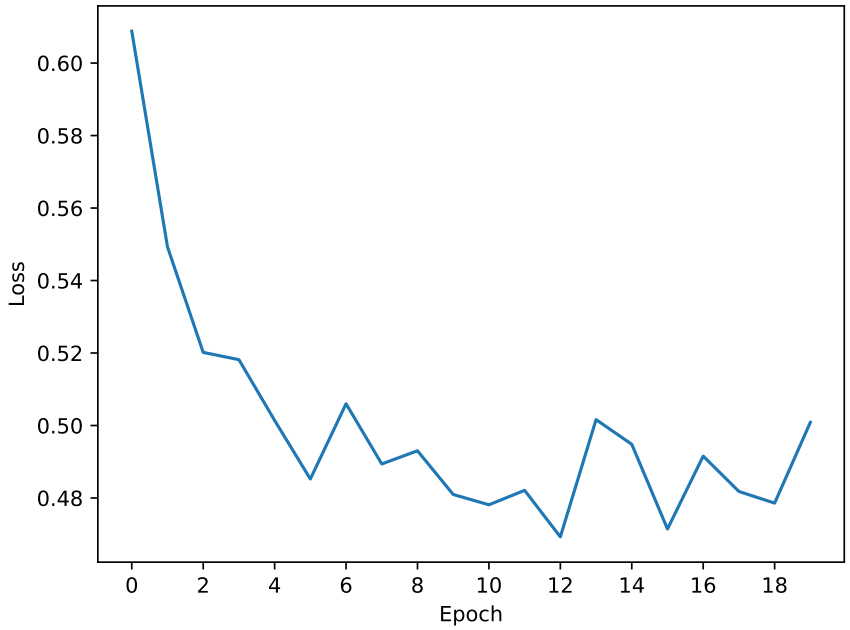
Τα αποτελέσματα είναι τα εξής:

- Training dataset:
  - Accuracy: 0.828
  - Recall: 0.828
  - F1 Score: 0.827
  - Loss:



- Test dataset:
  - Accuracy: 0.769
  - Recall: 0.454
  - F1 Score: 0.496

- Loss:



2.2

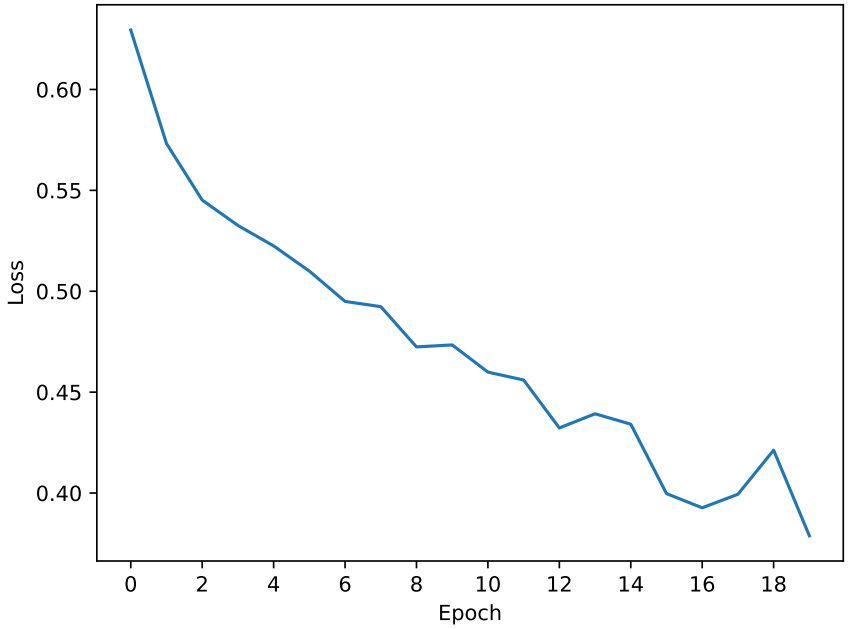
Εδώ χρησιμοποιούμε ως αναπαράσταση του κειμένου u την συνένωση:

$$u = [h_N||mean(E)||max(E)]$$

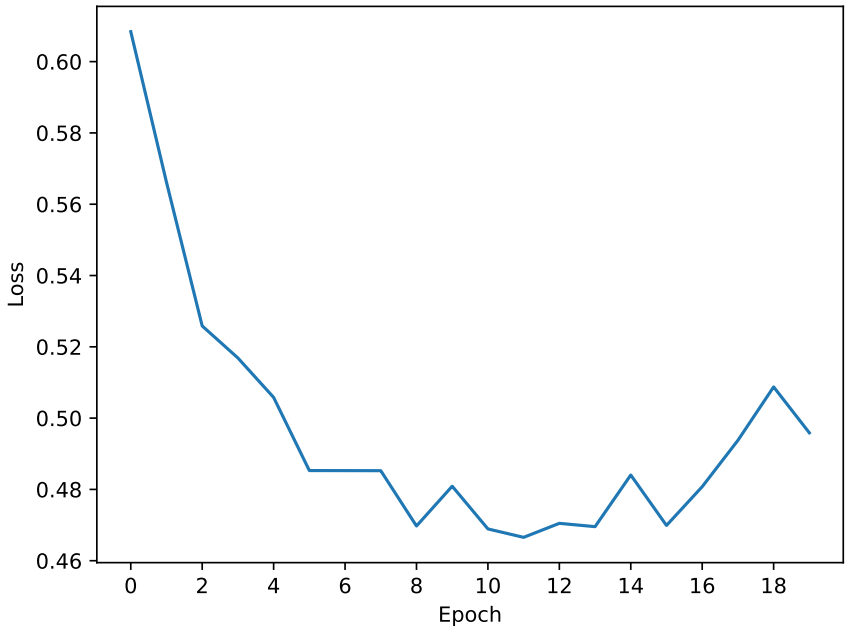
Η υλοποίηση βρίσκεται στην κλάση `Model22`, στο `models.py`.

Τα αποτελέσματα είναι τα εξής:

- Training dataset:
  - Accuracy: 0.840
  - Recall: 0.839
  - F1 score: 0.839
  - Loss:



- Test dataset:
  - Accuracy: 0.778
  - Recall: 0.482
  - F1 score: 0.503
  - Loss:



Ερώτημα 3

Σε αυτό το ερώτημα χρησιμοποιούμε ένα μηχανισμό attention. Παίρνουμε έτοιμη την υλοποίηση που βρίσκεται σε αυτό το gist.

3.1

Χρησιμοποιούμε τον μηχανισμό attention, για να υπολογίσουμε την αναπαράσταση ενός κειμένου, ως το σταθμισμένο άθροισμα των word embeddings:

$$v_i = tanh(We_i + b)$$

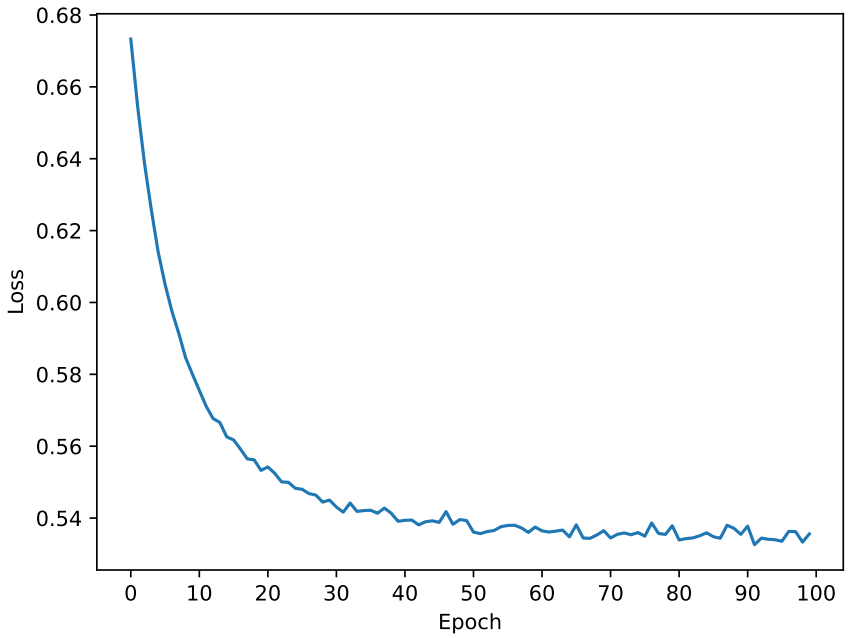
$$a_i = \frac{exp(v_i)}{\sum_{t=1}^{N} exp(u_t)}$$

$$u = \sum_{i=1}^{N} a_i e_i$$

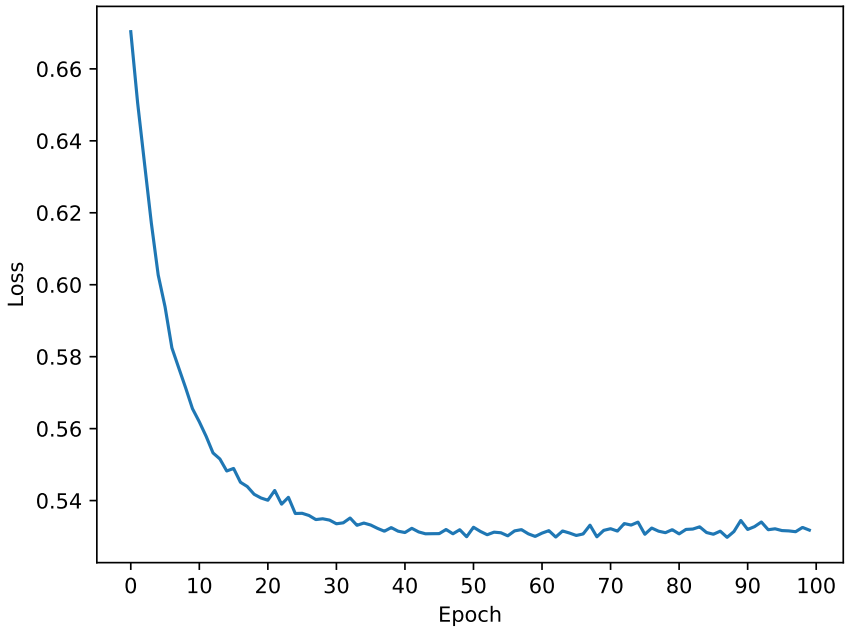
Η υλοποίηση βρίσκεται στην κλάση `Model31`, στο `models.py`.

Τα αποτελέσματα είναι τα εξής:

- Training dataset:
  - Accuracy: 0.726
  - Recall: 0.726
  - F1 Score: 0.724
  - Loss:



- Test dataset:
  - Accuracy: 0.734
  - Recall: 0.432
  - F1 Score: 0.478
  - Loss:



3.2

Πάλι χρησιμοποιούμε τον μηχανισμό attention για να υπολογίσουμε την αναπαράσταση ενός κειμένου, αλλά αυτή τη φορά ως το σταθμισμένο άθροισμα των εξόδων ενός LSTM.

$$v_i = tanh(Wh_i + b)$$

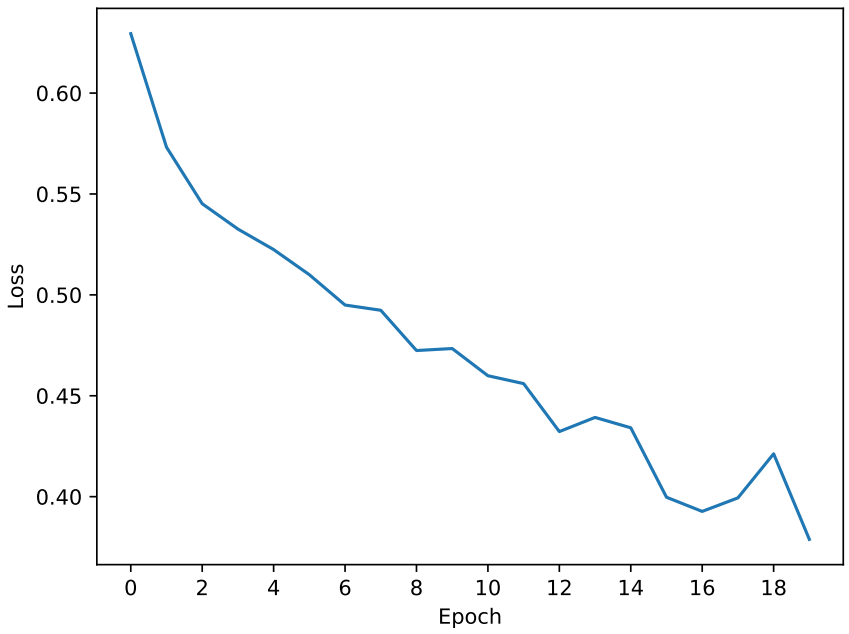
$$a_i = \frac{exp(v_i)}{\sum_{t=1}^{N} exp(u_t)}$$

$$u = \sum_{i=1}^{N} a_i h_i$$

Η υλοποίηση βρίσκεται στην κλάση `Model32`, στο `models.py`.

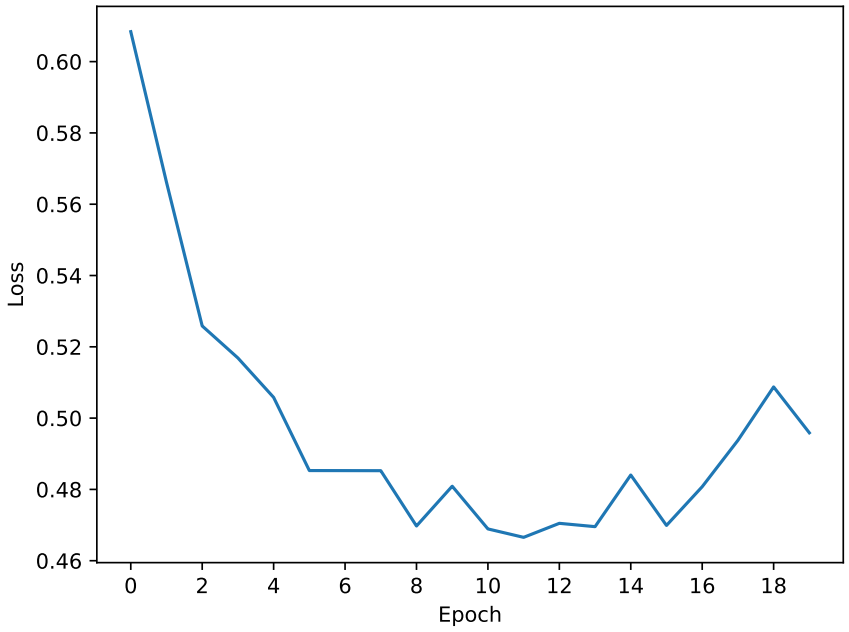
Τα αποτελέσματα είναι τα εξής:

- Training dataset:
  - Accuracy: 0.828
  - Recall: 0.828
  - F1 Score: 0.826
  - Loss:



- Test dataset:

- Accuracy: 0.745
- Recall: 0.438
- F1 Score: 0.485
- Loss:



Ερώτημα 4

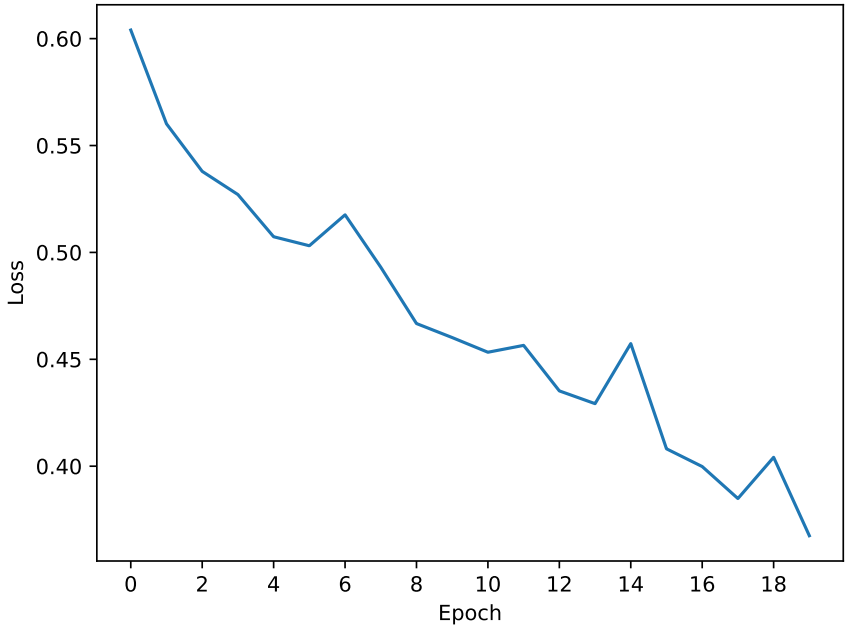
Σε αυτό το ερώτημα υλοποιούμε τα ζητούμενα του ερωτήματος 3, αλλά αυτή τη φορά χρησιμοποιούμε ένα **αμφίδρομο** LSTM.

4.1

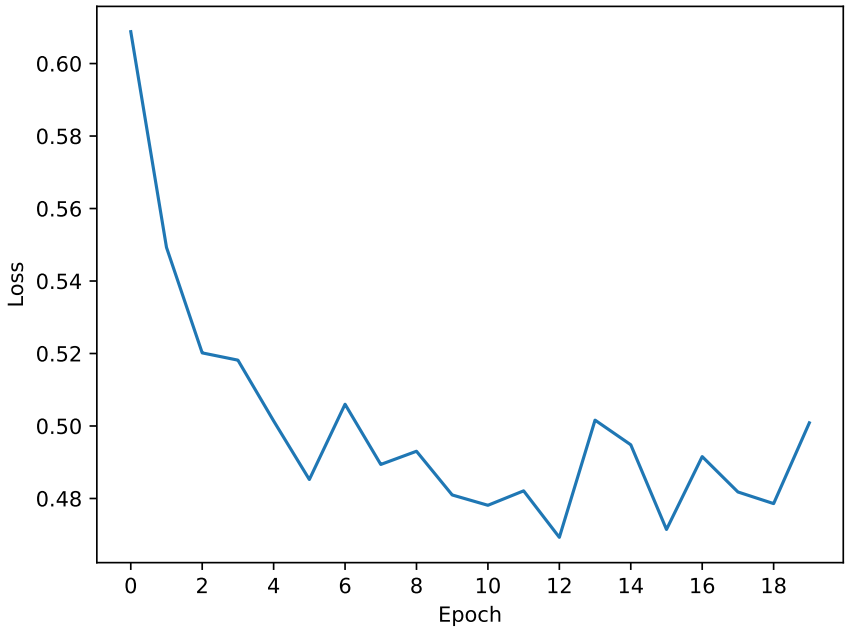
Αυτό το ερώτημα είναι όμοιο με το 2.2, αλλά με αμφίδρομο LSTM. Η υλοποίηση βρίσκεται στην κλάση `Model41`, στο `models.py`.

Τα αποτελέσματα είναι τα εξής:

- Training dataset:
  - Accuracy: 0.831
  - Recall: 0.832
  - F1 score: 0.828
  - Loss:



- Test dataset:
  - Accuracy: 0.782
  - Recall: 0.523
  - F1 score: 0.521
  - Loss:

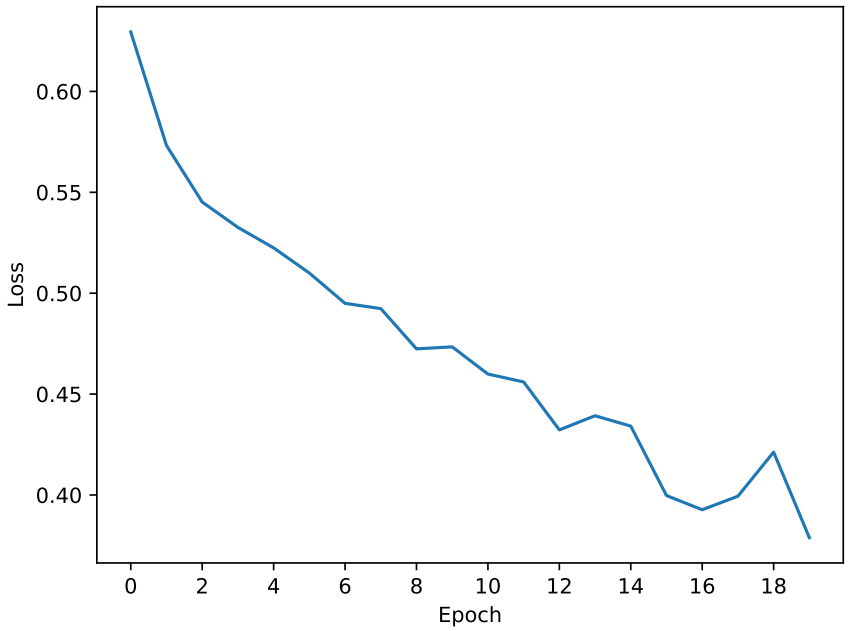


4.2

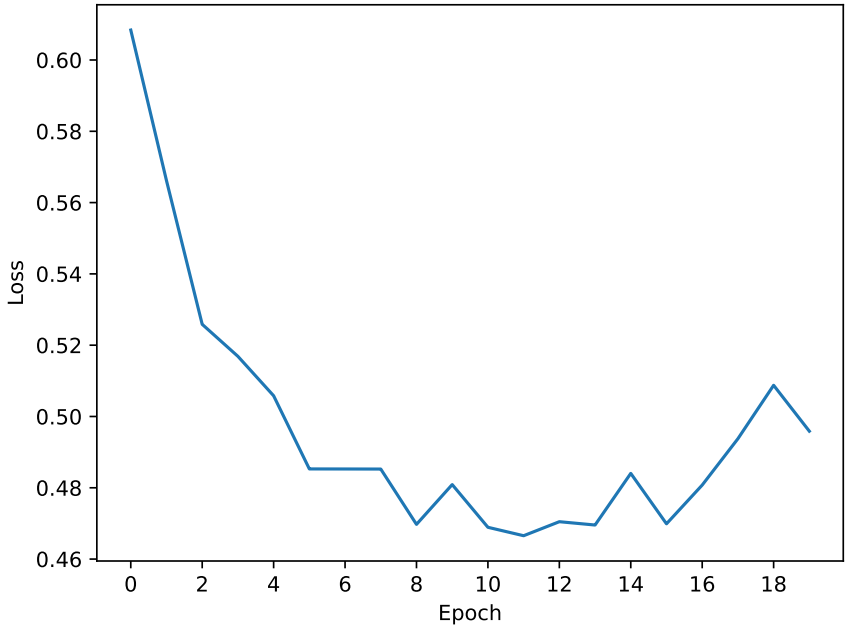
Αυτό το ερώτημα είναι όμοιο με το 3.2, αλλά με αμφίδρομο LSTM. Η υλοποίηση βρίσκεται στην κλάση `Model42`, στο `models.py`.

Τα αποτελέσματα είναι τα εξής:

- Training dataset:
  - Accuracy: 0.821
  - Recall: 0.819
  - F1 score: 0.819
  - Loss:



- Test dataset:
  - Accuracy: 0.790
  - Recall: 0.781
  - F1 score: 0.783
  - Loss:



Ερώτημα 5

5.1

...

5.2

...

5.3

...

Ερώτημα 6

6.1

...

6.2

Τα BoW χαρακτηριστικά θα μπορούσουν να οδηγήσουν σε καλύτερα αποτελέσματα σε σχέση με αναπαραστάσεις όπως ο μέσος όρος των word embeddings, σε περιπτώσεις όπου:

- Θέλουμε να δημιουργήσουμε baseline models, αφού πολύ εύκολα και γρήγορα, με έτοιμες συναρτήσεις βιβλιοθηκών, και χωρίς περίπλοκες αναπαραστάσεις, μπορούμε να φτιάξουμε μία απλοϊκή μορφή μοντέλων για κατηγοριοποίηση κειμένων.

- Το dataset είναι μικρό σε μέγεθος και έχουμε να κάνουμε με domain-specific context. Όταν τα κείμενα περιέχουν εξειδικευμένους/τεχνικούς όρους, τα pretrained embeddings που υπάρχουν δεν αποδίδουν τόσο καλά, γιατί είναι πολύ γενικά, και ταυτόχρονα είναι πιο δύσκολο να βρούμε τόσο εξειδικευμένα pretrained embeddings. Έτσι, προτιμάται η αναπαράσταση με χαρακτηριστικά tfidf.