



ΕΘΝΙΚΟ ΜΕΤΣΟΒΙΟ ΠΟΛΥΤΕΧΝΕΙΟ

Επεξεργασία Φωνής & Φυσικής Γλώσσας

Προπαρασκευή 3ο Εργαστηρίου

Γεωργίου Δημήτριος (03115106)

<el15106@central.ntua.gr>

Μπαζώτης Νικόλαος (03115739)

<el15739@central.ntua.gr>

Ιανουάριος 2019

1 Προεπεξεργασία δεδομένων

1.1 Κωδικοποίηση Επισημοποιήσεων (labels) - main.py

Στο συγκεκριμένο στάδιο γίνεται χρήση της *LabelEncoder()* της βιβλιοθήκης *sklearn*, με σκοπό την μετατροπή των labels από μορφή string (και συγκεκριμένα positive, negative και neutral για το αρχείο **Semeval2017A**, ενώ positive, negative για το αρχείο **MR**) αριθμούς, ανάλογα να μετατραπούν σε μορφή κατάλληλη για την εισαγωγή τους στο νευρωνικό.

Ζητούμενο 1:

1. Εκτύπωση τα labels και τα τις κωδικοποιημένες αντιστοιχίες τους για το αρχείο **Semeval2017A**

```
#####EX1#####
neutral 1
positive 2
neutral 1
positive 2
positive 2
positive 2
neutral 1
positive 2
negative 0
neutral 1
```

2. Εκτύπωση τα labels και τα τις κωδικοποιημένες αντιστοιχίες τους για το αρχείο **MR**

```
#####EX1#####
positive 1
positive 1
positive 1
positive 1
positive 1
positive 1
positive 1
positive 1
positive 1
positive 1
```

1.2 Λεκτική Ανάλυση (Tokenization) - main.py, dataloading.py

Στο στάδιο αυτό γίνεται μετατροπή των δεδομένων-προτάσεων που υπάρχουν στα δύο αρχεία σε ακολουθίες από tokens. Πιο συγκεκριμένα στην συνάρτηση αρχικοποίησης *__init__()* της κλάσης *SentenceDataset*, αρχικοποιούμε τις μεταβλητές *data*, *labels*, *word2idx*, *avg_length* με τα tokens, τα string labels, το λεξικό των embeddings και το κατά μέσο όρο μήκος για τις προτάσεις του συγκεκριμένου κειμένου (θα επεξηθεί παρακάτω).

Για την διαδικασία του tokenization: Ορίσαμε δύο customized συναρτήσεις *parser()* και *tokenize()*, όπως αυτές είχαν οριστεί στο εργαστήριο lab1. Η *tokenize()*, δέχεται ως όρισμα ένα string και κάνει process αυτού, μετατρέποντας κεφαλαία σε πεζά, διατηρώντας μόνο αλφαριθμητικούς χαρακτήρες, διασπώντας σε μία λίστα από λέξεις, ενώ χρησιμοποιεί την *" ".join(string.split())* για εξάλειψη διπλότυπων κενών ώστε στην συνέχεια η εφαρμογή της *split()* να επιστρέφει μόνο αυτές σε μορφή λίστας (και όχι extra κενά ως λέξεις). Η *parser()*, δέχεται ως όρισμα όλο το αρχείο προτάσεων του **Semeval2017A** και του **MR**, και τα διατρέχει γραμμή - γραμμή, εφαρμόζοντας στο string κάθε γραμμής την συνάρτηση *tokenize()*, παράγοντας τα αντίστοιχα tokens και αποθηκεύοντας τα σε μία λίστα από λίστες (όπου λίστα = tokens πρότασης).

Σημείωση: Για το αρχείο **MR** το tokenization είναι θεωρητικά decent και οδηγεί σε καλά αποτελέσματα και accuracy μοντέλου, το οποίο οφείλεται στην ήδη καλή μορφή του κειμένου. Από την άλλη το **Semeval2017A** που περιέχει tweets, αποτελεί ανασταλτικό παράγοντα για την αποδοτική λειτουργία του customized tokenization μας, οπότε παρακάτω γίνεται ανάλυση τόσο για χρήση αυτού όσο και για χρήση του *TweetTokenizer()* της βιβλιοθήκης *nlTK*.

Παρακάτω φαίνεται η χρήση τους για τις 10 πρώτες προτάσεις για κάθε dataset

Ζητούμενο 2:

1. Τα tokens και το αντίστοιχο κωδικοποιημένο label για το αρχείο **Semeval2017A** με χρήση του customized tokenization είναι:

```
#####EX2#####
['beat', 'it', 'michael', 'jackson', 'thriller', 'th', 'anniversary', 'edition', 'hd'] 1
['jay', 'z', 'joins', 'instagram', 'with', 'nostalgic', 'tribute', 'to', 'michael', 'jackson', 'jay', 'z', 'apparently', 'joined', 'instagram',
 'on', 'saturday', 'and'] 2
['michael', 'jackson', 'bad', 'th', 'anniversary', 'edition', 'picture', 'vinyl', 'this', 'unique', 'picture', 'disc', 'vinyl', 'includes', 'th
e', 'original'] 1
['i', 'liked', 'a', 'youtube', 'video', 'one', 'direction', 'singing', 'man', 'in', 'the', 'mirror', 'by', 'michael', 'jackson', 'in', 'atlanta
', 'ga', 'june'] 2
['th', 'anniv', 'of', 'princess', 'dianas', 'death', 'i', 'still', 'want', 'to', 'believe', 'she', 'is', 'living', 'on', 'a', 'private', 'islan
d', 'away', 'from', 'the', 'public', 'with', 'michael', 'jackson'] 2
['oridaganjazz', 'the', 'st', 'time', 'i', 'heard', 'michael', 'jackson', 'sing', 'was', 'in', 'honolulu', 'hawaii', 'a', 'restaurant', 'on', '
radio', 'it', 'was', 'abc', 'i', 'was', 'i', 'loved', 'it'] 2
['michael', 'jackson', 'appeared', 'on', 'saturday', 'at', 'the', 'th', 'place', 'in', 'the', 'top', 'of', 'miamis', 'trends', 'trndnl'] 1
['are', 'you', 'old', 'enough', 'to', 'remember', 'michael', 'jackson', 'attending', 'the', 'grammys', 'with', 'brooke', 'shields', 'and', 'web
ster', 'sat', 'on', 'his', 'lap', 'during', 'the', 'show'] 2
['etbowser', 'do', 'u', 'enjoy', 'his', 'nd', 'rate', 'michael', 'jackson', 'bit', 'honest', 'ques', 'like', 'the', 'cant', 'feel', 'face', 'so
ng', 'but', 'god', 'its', 'so', 'obvious', 'they', 'want', 'mj'] 0
['the', 'weeknd', 'is', 'the', 'closest', 'thing', 'we', 'may', 'get', 'to', 'michael', 'jackson', 'for', 'a', 'long', 'timespecially', 'since
', 'he', 'damn', 'near', 'mimics', 'everything'] 1
#####
```

- Τα tokens και το αντίστοιχο κωδικοποιημένο label για το αρχείο **MR** με χρήση του customized tokenization είναι:

```
#####EX2#####
['the', 'rock', 'is', 'destined', 'to', 'be', 'the', 'st', 'centurys', 'new', 'conan', 'and', 'that', 'hes', 'going', 'to', 'make', 'a', 'splash',
 'even', 'greater', 'than', 'arnold', 'schwarzenegger', 'jeanclaud', 'van', 'damme', 'or', 'steven', 'segal'] 1
['the', 'gorgeously', 'elaborate', 'continuation', 'of', 'the', 'lord', 'of', 'the', 'rings', 'trilogy', 'is', 'so', 'huge', 'that', 'a', 'column',
 'of', 'words', 'cannot', 'adequately', 'describe', 'cowriterdirector', 'peter', 'jacksons', 'expanded', 'vision', 'of', 'j', 'r', 'r', 'tolkiens',
 'middleearth'] 1
['effective', 'but', 'tootepid', 'biopic'] 1
['if', 'you', 'sometimes', 'like', 'to', 'go', 'to', 'the', 'movies', 'to', 'have', 'fun', 'wasabi', 'is', 'a', 'good', 'place', 'to', 'start'] 1
['emerges', 'as', 'something', 'rare', 'an', 'issue', 'movie', 'thats', 'so', 'honest', 'and', 'keenly', 'observed', 'that', 'it', 'doesnt', 'feel
', 'like', 'one'] 1
['the', 'film', 'provides', 'some', 'great', 'insight', 'into', 'the', 'neurotic', 'mindset', 'of', 'all', 'comics', 'even', 'those', 'who', 'have
', 'reached', 'the', 'absolute', 'top', 'of', 'the', 'game'] 1
['offers', 'that', 'rare', 'combination', 'of', 'entertainment', 'and', 'education'] 1
['perhaps', 'no', 'picture', 'ever', 'made', 'has', 'more', 'literally', 'showed', 'that', 'the', 'road', 'to', 'hell', 'is', 'paved', 'with', 'goo
d', 'intentions'] 1
['steers', 'turns', 'in', 'a', 'snappy', 'screenplay', 'that', 'curls', 'at', 'the', 'edges', 'its', 'so', 'clever', 'you', 'want', 'to', 'hate', '
it', 'but', 'he', 'somehow', 'pulls', 'it', 'off'] 1
['take', 'care', 'of', 'my', 'cat', 'offers', 'a', 'refreshingly', 'different', 'slice', 'of', 'asian', 'cinema'] 1
```

1.3 Κωδικοποίηση Παραδειγμάτων (λέξεων) - dataloading.py, main.py

Στο στάδιο αυτό γίνεται υλοποίηση της συνάρτησης `__getitem__()` της κλάσης `SentenceDataset`, η οποία δέχεται ως όρισμα έναν αριθμό `index` και επιστρέφει τα παρακάτω:

- Την κωδικοποιημένη μορφή μιάς πρότασης, με το αντίστοιχο zero-padding το οποίο γίνεται ως εξής: Στην **main.py**:
 - Έχουμε ορίσει την συνάρτηση `reject_outliers()`, η οποία δέχεται ως όρισμα το numpy array των προτάσεων του αντίστοιχου κειμένου και το επιστρέφει χωρίς τα outliers τα οποία μπορεί να επηρέαζαν αρκετά μέσο όρο στα μήκη των προτάσεων,
 - Για κάθε αρχείο και για κάθε dataset (train, test), υπολογίζουμε το κατά μέσο όρο μήκος προτάσεων `avg_length_without_outliers`, το οποίο περνάμε ως όρισμα στα `train_set`, `test_set` που ορίσαμε στο βήμα 1.2
 - Έτσι λοιπόν χρησιμοποιώντας το μήκος αυτό ως threshold, τρέχουμε επαναληπτικά τις λέξεις μιας πρότασης υπολογίζοντας την κωδικοποίηση τους μέσω του λεξικού `word2idx`¹, την οποία επεκτείνουμε με μηδενικά αν το μήκος της είναι μικρότερο από το threshold αυτό, αλλιώς τις κόβουμε τις extra λέξεις σε αντίθετη περίπτωση. Αυτό γίνεται διότι και τα δύο dataset **πρέπει** πρέπει να έχουν προτάσεις με ίδιο μήκος, ώστε να εφαρμοστούν με επιτυχία στο `Dataloader()`
 Σημειώνεται ότι: παρόλο που χρησιμοποιούμε το average ως κοινό παρανομαστή μήκους, θα μπορούσαμε να βρούμε εναλλακτικό μήκος που καλύπτει την πλειοψηφία των προτάσεων, αγνοώντας εκ νέου τους outliers
- Το id της αντίστοιχης υποσημείωσης label
- Το πραγματικό μήκος της πρότασης, με την έννοια της εξαίρεσης των μηδενικών στοιχείων.

Ζητούμενο 3: Παρακάτω φαίνεται η χρήση της `__getitem__()` για τις 5 πρώτες προτάσεις για κάθε αρχείο

- Εκτύπωση αρχικής και αντίστοιχης επεξεργασμένης μορφής για το αρχείο **Semeval2017A**

¹ Αν δεν υπάρχει η λέξη στο λεξικό, παίρνουμε την κωδικοποίηση της λέξης `unk`

```
#####EX3#####
calculating average length with and without outliers...
Average length with outliers is: 18
Average length without outliers is: 19
printing 5 word embeddings in the original and the transformed form using average length...
WORD EMBEDDING 0
sentence: 05 Beat it - Michael Jackson - Thriller (25th Anniversary Edition) [HD] http://t.co/A4K2B86PBv , target: neutral
sentence's word embedding: [ 961 21 786 1755 8966 14359 2350 2493 12315 0 0 0
0 0 0 0 0 0 0] , label: 1

WORD EMBEDDING 1
sentence: Jay Z joins Instagram with nostalgic tribute to Michael Jackson: Jay Z apparently joined Instagram on Saturday and.. http://t.co/Qj9I4eCvXy , target: positive
sentence's word embedding: [ 4791 9027 7698 109263 18 20557 5079 5 786 1755
4791 9027 1897 1031 109263 14 278 6 0] , label: 2

WORD EMBEDDING 2
sentence: Michael Jackson: Bad 25th Anniversary Edition (Picture Vinyl): This unique picture disc vinyl includes the original 1 http://t.co/fKXhToAAuW , target: neutral
sentence's word embedding: [ 786 1755 979 14359 2350 2493 1836 11193 38 3007 1836 5977
11193 1013 1 930 0 0 0] , label: 1

WORD EMBEDDING 3
sentence: I liked a @YouTube video http://t.co/AaR3pjp2PI One Direction singing "Man in the Mirror" by Michael Jackson in Atlanta, GA [June 26, , target: positive
sentence's word embedding: [ 42 5573 8 8365 975 49 2192 4100 301 7 1 6462
22 786 1755 7 1098 12132 345] , label: 2

WORD EMBEDDING 4
sentence: 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. , target: positive
sentence's word embedding: [ 14359 400001 4 4243 385972 337 42 150 304 5
734 68 15 757 14 8 673 584 421] , label: 2
```

2. Εκτύπωση αρχικής και αντίστοιχης επεξεργασμένης μορφής για το αρχείο MR

```
#####EX3#####
calculating average length with and without outliers...
Average length with outliers is: 18
Average length without outliers is: 17
printing 5 word embeddings in the original and the transformed form using average length...
WORD EMBEDDING 0
sentence: 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 . , target: positive
sentence's word embedding: [ 1 1138 15 10454 5 31 1 2697 400001 51
18513 6 13 69408 223 5 160] , label: 1

WORD EMBEDDING 1
sentence: the gorgeously elaborate continuation of " the lord of the rings " trilogy is so huge that a column of words cannot adequately describe co-writer/director peter jackson's expanded vision of j . r . r . tolkien's middle-earth . , target: positive
sentence's word embedding: [ 1 78616 5135 10117 4 1 2371 4 1 6820 12305 15
101 1325 13 8 3236] , label: 1

WORD EMBEDDING 2
sentence: effective but too-tepid biopic , target: positive
sentence's word embedding: [ 2038 35 400001 34277 0 0 0 0 0 0
0 0 0 0 0 0] , label: 1

WORD EMBEDDING 3
sentence: if you sometimes like to go to the movies to have fun , wasabi is a good place to start . , target: positive
sentence's word embedding: [ 84 82 1072 118 5 243 5 1 2460 5 34 2906
66408 15 8 220 242] , label: 1

WORD EMBEDDING 4
sentence: emerges as something rare , an issue movie that's so honest and keenly observed that it doesn't feel like one . , target: positive
sentence's word embedding: [ 12398 20 646 2349 30 496 1006 86909 101 6082
6 23499 4583 13 21 136284 999] , label: 1
```

2 Μοντέλο - models.py

Στην συγκεκριμένη ενότητα χτίζουμε την βασική δομή και τα layers ενός νευρωνικού. Η μέθοδος `__init__()` της κλάσης `BaselineDNN` χρησιμοποιείται για την δήλωση των layers του δικτύου καθώς για την αρχικοποίηση των βαρών τους.

2.1 Embedding Layer

Στο συγκεκριμένο στάδιο δημιουργούμε το **Embedding Layer**, το οποίο με όρισμα, τις προτάσεις του κειμένου - που πλέον έχουν όλες ίδιο μήκος και οι οποίες βρίσκονται σε μορφή `dataloader` - κάνει προβολή αυτών στον συνεχή χώρο, στον οποίο απέχουν μικρή απόσταση οι νοηματικά κοντινές λέξεις.

Γενικότερα για τα βάρη του layer διαθέτουμε δύο επιλογές. Είτε να τα αρχικοποιήσουμε με τυχαίες τιμές -με αντίστοιχη ενημέρωση αυτών αργότερα κατά την εκπαίδευση του μοντέλου- είτε να τα αρχικοποιήσουμε με τα pretrained word embeddings. Για την παρούσα προπαρασκευή, γίνεται αρχικοποίηση αυτών με τα pretrained word embeddings, όπως αυτά έχουν προκύψει από αντίστοιχο dataset **glove.6B.50d.txt**, δηλαδή κάνουμε χρήση των 50-dimensional Glove Embeddings.

Για το layer ορίζουμε το `num_embeddings` = μέγεθος του παραπάνω αρχείου, `embedding_dim`= διάσταση των embeddings 50

Ζητούμενο 4:

- **Ερώτημα 1:** Γιατί αρχικοποιούμε το *embedding layer* με τα προ-εκπαιδευμένα *word embeddings*?

Χρησιμοποιούμε τα προ-εκπαιδευμένα word embeddings διότι θέλουμε να διευκολύνουμε το νευρωνικό δίκτυο να κάνει την εκπαίδευση του καλύτερα και πιο γρήγορα πατώντας πάνω σε ήδη εκπαιδευμένα word-embeddings τα οποία αναπαριστούν τις σημασιολογικά κοντινές λέξεις κοντά στον πολυδιάστατο χώρο.

Αντιθέτως αν είχαμε τυχαίες τιμές και τις αφήναμε να εκπαιδευτούν πάνω στα δεδομένα μας θα είχαμε αρχικά αναπαραστάσεις οι οποίες θα ήταν πολύ μακριά σε προσέγγιση με αυτό που επιθυμούμε ,δηλαδή τυχαία αναπαράσταση των λέξεων στον χώρο . Επίσης οι τυχαίες τιμές θα είχαν το μειονέκτημα ότι για να εκπαιδευτεί με αυτές το νευρωνικό δίκτυο θα αργούσε να εξάγει την σωστή αναπαράσταση και κατά συνέπεια η ακρίβεια του θα επηρεαζόταν επίσης .

- **Ερώτημα 2:** Γιατί κρατάμε παγωμένα τα βάρη του *embedding layer* κατά την εκπαίδευση?

Τα παγωμένα βάρη του embedding layer είναι ουσιαστικά οι παράμετροι τις οποίες αποτρέπουμε από το να τροποποιηθούν περαιτέρω. Αυτή η τεχνική χρησιμοποιείται ιδιαίτερα στο transfer learning , όπου το base model (εκπαιδευμένο σε άλλο dataset) είναι παγωμένο.

Η συνεισφορά αυτής της τεχνικής είναι σημαντική στην ταχύτητα εκτέλεσης της εκπαίδευσης. Είναι προφανές πως εάν δεν θέλουμε να τροποποιηθούν οι παράμετροι μας αυτό υποδηλώνει την μη διεξαγωγή του backward pass οδηγώντας σε σημαντική αύξηση της ταχύτητας σχεδόν στο διπλάσιο καθώς η μισή διαδικασία (βλ. backward pass) παραλείπεται.

Επιπλέον αξίζει να αναφερθεί ότι στην περίπτωση μας η τεχνική αυτή εφαρμόζεται γιατί έχουμε προ-εκπαιδευμένα word-embeddings και εκτός του κέρδους στην ταχύτητα ελλοχεύει ο κίνδυνος του overfitting δηλαδή ο κίνδυνος το μοντέλο μας ουσιαστικά να μάθει πολύ καλά τα δεδομένα που του δώσαμε αν εκτός από το νευρωνικό είχαμε και τις αναπαραστάσεις που μπαίνουν σε αυτό να εκπαιδεύονται πάνω στο **train_set** μας

2.2 Output Layer

Στο συγκεκριμένο στάδιο κάνουμε προβολή των αναπαραστάσεων στον χώρο των κλάσεων. Αξίζει να σημειωθεί πως έγινε επιλογή μεγέθους `hiddensize` = 128, το οποίο υποδηλώνει ότι οι τελικές αναπαραστάσεις είναι 128 διαστάσεων και πρόβλημα της κατηγοριοποίησης έχει 3 και 2 κλάσεις αντίστοιχα για τα κείμενα **Semeval2017A** και **MR**. Επομένως το τελευταίο layer μας θα κάνει προβολή $R^{128} \rightarrow R^{3_{Semeval2017A}}, R^{2_{MR}}$.

Αξίζει να σημειωθεί πως το output layer κατασκευάζεται σε δύο στάδια:

1. Διαθέτουμε ένα layer με μία μη γραμμική συνάρτηση ενεργόποιησης (`relu()`, `tanh()`) - θα γίνουν δοκιμές και με τα δύο -. Αυτό μετασχηματίζει
2. Τελευταίο τελικό layer το οποίο είναι υπεύθυνο για την προβολή αυτή των τελικών αναπαραστάσεων στον χώρο των κλάσεων

Ζητούμενο 5:

- **Ερώτημα 3:** Γιατί βάζουμε μία μη γραμμική συνάρτηση ενεργοποίησης στο προτελευταίο layer? Τι διαφορά θα είχε αν είχαμε 2 ή περισσότερους γραμμικούς μετασχηματισμούς στην σειρά?

Προτιμάται να έχουμε μια μη γραμμική συνάρτηση ενεργοποίησης διότι έχουν βαθμό >1 και η αναπαράσταση τους δεν είναι ευθεία γραμμή αλλά παρουσιάζει κυρτότητα. Αυτό αποδεικνύεται πολύ χρήσιμο διότι όταν έχουμε ένα νευρωνικό δίκτυο το οποίο θέλει να αναπαραστήσει σχεδόν τα πάντα και με καλή ακρίβεια χρειαζόμαστε κάτι πιο πολύπλοκο από μια γραμμική μπορεί να αντιστοιχίσει με μεγαλύτερη επιτυχία την είσοδο με την έξοδο.

Αντίθετα έχοντας 2 γραμμικούς μετασχηματισμούς στην σειρά εξακολουθούμε να έχουμε συνολικά γραμμική συνάρτηση διότι το γινόμενο γραμμικών μετασχηματισμών είναι γραμμικός μετασχηματισμός και έτσι το μοντέλο έχοντας γραμμική συνάρτηση ενεργοποίησης στο προτελευταίο layer έχει ένα πολύ περιορισμένο εύρος το οποίο μπορεί να αντιστοιχίσει καλά από είσοδο σε έξοδο.

2.3 Forward pass

Στο συγκεκριμένο στάδιο θα γίνει forward propagation. Θα δομήσουμε τον τρόπο μετασχηματισμού μιας εισόδου - χειμένου που του δίνουμε στις αντίστοιχες εξόδους. Μάλιστα, μια εποχή ισοδυναμεί με ένα forward pass (λαμβάνουμε τις output τιμές) και ένα backward pass (ενημέρωση βαρών) ισοδυναμεί με όλα τα training batches.

Γενικότερα το να επιτρέπουμε σε ένα backward pass για ένα minibatch (γιατί εργαζόμαστε με minibatches) να χρησιμοποιεί περισσότερες ενημερωμένες παραμέτρους από αυτές που χρησιμοποιούνται στο αντίστοιχο backward pass είναι μεγάλο πρόβλημα.

Αφού έχει ολοκληρωθεί το forward pass για ένα minibatch, κάθε στάδιο-layer στέλνει ασύγχρονα τα output activations στο επόμενο stage-layer ενώ ταυτόχρονα ξεκινάει η διαδικασία processing άλλου minibatch. Συγκεκριμένα για την περίπτωση μας, η είσοδος στο μοντέλο είναι ένα minibatch με διαστάσεις (batch_size, max_length), ενώ η έξοδος των layers έχει ως εξής:²

- Οι λέξεις κάθε πρότασης διοχετεύονται στο **embedding layer**, ώστε να γίνει αντιστοίχιση κάθε όρου σε ένα διάνυσμα, συνολικά η έξοδος του layer θα έχει διαστάσεις (batch_size, max_length, emd_dim).
- Δημιουργούμε τις επιμέρους αναπαραστάσεις των όρων, συγκεκριμένα μία ενιαία για κάθε πρόταση, υπολογίζοντας των μέσο όρο των embeddings σε μία πρόταση, κάνοντας χρήση της customized συνάρτησης `mean_pooling()`
- Εφαρμόζουμε τον μην μη γραμμικό μετασχηματισμό στην παραπάνω αναπαράσταση
- Γίνεται τελική προβολή στον χώρο των κλάσεων

Σημειώνεται ότι το παραπάνω περιλαμβάνει πολλές επιμέρους λειτουργίες, με αποτέλεσμα να απαιτούν μεγάλη προσοχή στην υλοποίησή τους. Για ένα minibatch, παρουσιάζουμε τα αποτελέσματα παρακάτω:

²Η διαδικασία κάνει apply και για τα δύο αρχεία

```

Step1
Size of the input in the EMBEDDING LAYER: torch.Size([128, 19])
tensor([ 160, 1086, 5, 2376, 393, 1100, 84, 211667, 882,
        14036, 26, 660, 494, 11989, 3078, 5, 4976, 102856,
        0])

Step2
Size of the output of the EMBEDDING LAYER: torch.Size([128, 19, 50])

Step3
Size of the new representation of the output of the EMBEDDING LAYER: torch.Size([128, 50])
tensor([ 0.1656, 0.0327, 0.0917, -0.1859, 0.1842, 0.0897, -0.4184, 0.0056,
        0.0716, 0.1132, -0.1195, 0.3096, 0.0638, -0.1009, 0.4839, 0.1572,
        -0.0770, -0.2220, -0.0050, -0.2667, 0.2096, 0.0907, 0.3623, 0.2392,
        0.2338, -1.3355, -0.4159, -0.0196, 0.5143, -0.2567, 2.4464, 0.3877,
        -0.3146, -0.2002, -0.1031, 0.1207, -0.0555, -0.0723, 0.1979, -0.1322,
        0.2268, 0.2716, -0.0457, 0.2697, 0.0570, 0.0441, 0.0071, 0.4322,
        0.2549, 0.2297])

Step4
Size of the output of the ACTIVATION LAYER: torch.Size([128, 50])
tensor([0.1656, 0.0327, 0.0917, 0.0000, 0.1842, 0.0897, 0.0000, 0.0056, 0.0716,
        0.1132, 0.0000, 0.3096, 0.0638, 0.0000, 0.4839, 0.1572, 0.0000, 0.0000,
        0.0000, 0.0000, 0.2096, 0.0907, 0.3623, 0.2392, 0.2338, 0.0000, 0.0000,
        0.0000, 0.5143, 0.0000, 2.4464, 0.3877, 0.0000, 0.0000, 0.0000, 0.1207,
        0.0000, 0.0000, 0.1979, 0.0000, 0.2268, 0.2716, 0.0000, 0.2697, 0.0570,
        0.0441, 0.0071, 0.4322, 0.2549, 0.2297])

Step5
Size of the output of the FINAL CLASSIFIER LAYER: torch.Size([128, 3])
tensor([-0.6371, 0.3987, 0.1955])

```

Ζητούμενο 6:

- **Ερώτημα 4:** Αν θεωρήσουμε κάθε διάσταση του *embedding* χώρου αντιστοιχεί σε μια αφηρημένη έννοια, μπορείτε να δώσετε μια διαισθητική ερμηνεία για το τι περιγράφει η αναπαράσταση που φτιάξατε (κέντρο-βάρους)?

Σε προηγούμενο ερώτημα αναφέρθηκε ότι ο τρόπος που έχουμε επιλέξει να αναπαρίστανται οι λέξεις στο χώρο είναι τέτοιος ώστε λέξεις με σημασιολογική κοντινή σχέση μεταξύ τους να είναι και κοντά στον πολυδιάστατο χώρο.

Συνεπώς οι προτάσεις οι οποίες αποτελούνται από λέξεις (διανύσματα) στον πολυδιάστατο χώρο συνιστούν επίσης ένα διάνυσμα το οποίο σε κάθε διάσταση έχει τιμή των μέσο όρο των τιμών των λέξεων που την απαρτίζουν στην εκάστοτε διάσταση.

Επακόλουθο αυτού είναι το γεγονός πως μπορούμε να δούμε την σχέση που έχει η κάθε πρόταση με την συγκεκριμένη αφηρημένη έννοια απλά κοιτώντας την τιμή της σε αυτήν την διάσταση. Ωστόσο η αναπαράσταση αυτή δεν λαμβάνει υπόψιν της τα συντακτικά χαρακτηριστικά και έτσι υπάρχει το ενδεχόμενο να μην είναι απόλυτα έγκυρη η αποτύπωση των εννοιών που εμπεριέχεται σε μια πρόταση απλά και μόνο αναλύοντας τις επιμέρους λέξεις.

- **Ερώτημα 5:** Αναφέρετε πιθανές αδυναμίες της συγκεκριμένης προσέγγισης για να αναπαραστήσουμε κείμενα

Σημαντική αδυναμία όπως αναφέρθηκε είναι η σύνταξη της πρότασης η οποία δεν λαμβάνεται υπόψιν παρόλο που παίζει κυρίαρχο ρόλο στην αποτύπωση των εννοιών που διατυπώνονται πχ *book that flight*, εδώ μπορεί να συσχετιστεί το *book* ως “noun” και όχι ως “verb” κάτι το οποίο αλλάζει εξολοκλήρου το νόημα της πρότασης. Στην ίδια κλίμακα εντάσσεται και η σειρά εμφάνισης των λέξεων που παίζει ρόλο στην σημασία της πρότασης, για παράδειγμα *Dog loves Jim – Jim loves dog/*

Επίσης τα σημεία στίξης αφαιρούνται κατά την προετοιμασία των δεδομένων ωστόσο είναι και αυτά σημαντικός παράγοντας στην σημασία της πρότασης πχ *I hate that you owe me – I hate that, you owe me*.

3 Διαδικασία Εκπαίδευσης - *main.py*, *training.py*

Στην συγκεκριμένη ενότητα υλοποιούμε την διαδικασία εκπαίδευσης του δικτύου. Επίσης γίνεται οργάνωση των παραδειγμάτων σε mini-batches και εκτέλεση stochastic gradient descent για ενημέρωση βαρών του δικτύου.

3.1 Φόρτωση Παραδειγμάτων (Dataloaders) - main.py

Γίνεται χρήση της κλάσης `Dataloader()`, με την οποία διαχωρίζουμε το dataset που της έχουμε δώσει ως όρισμα σε mini batches. Γενικότερα μας επιτρέπει λειτουργίες πάνω σε set δεδομένων.

Αξίζει να σημειωθεί πως παρά το γεγονός ότι το βήμα αυτό είναι δυο γραμμές κώδικας, είναι μεγάλης σημαντικότητας, ώστε τα αρχεία κειμένου **Semeval2017A** και **MR** να οργανωθούν όσο το δυνατόν καλύτερα, μάλιστα αν τα datasets της 1η ενότητας δεν είναι στην επιθυμούσα μορφή, δεν προκύπτει απαραίτητα error το οποίο είναι παρ'όλα αυτά καταστροφικό για τα επόμενα βήματα (πχ. αποτυχία στην ενημέρωση βαρών)

- **Ερώτημα 6:** Τι συνέπειες έχουν τα μικρά και μεγάλα mini-batches στην εκπαίδευση των μοντέλων?

Αρχικά το batch size ορίζει τον αριθμό των δειγμάτων που θα μεταφερθούν μέσα στο network σε κάθε επανάληψη. Επίσης είναι αναμενόμενο ότι όσο μεγαλύτερο είναι το batch size, τόσο μικρότερο είναι το learning rate που απαιτείται προκειμένου να εκπαιδύσουμε με ακρίβεια. Γενικότερα το μέγεθος του mini batch επηρεάζει με δύο τρόπους

Κατά την διάρκεια εκτέλεσης του stochastic gradient descent, όσο μεγαλύτερο mini-batch έχουμε, τόσο λιγότερο θόρυβο διαθέτει το μοντέλο-σύστημα μας, καθώς βρισκόμαστε πιο κοντά στα gradients του training set. Αν και από μια πρώτη οπτική αυτό φαίνεται ξεκάθαρα θετικό, διαπιστώνεται ότι αν το mini-batch αυξηθεί αρκετά τότε το νευρωνικό θα μπορούσε να συγκλίνει σε κάποιο τοπικό ελάχιστο - και όχι στο ζητούμενο ολικό -, το οποίο θα μπορούσε να είχε αποφευχθεί από ένα νευρωνικό με μικρότερο μέγεθος batches

Αντιθέτως όσο μικρότερα batches έχουμε τόσο πιο γρήγορα γίνεται το update των βαρών σε κάθε επανάληψη, και αποκτούμε μια λίγο πιο ολοκληρωμένη εικόνα για το τι γίνεται στο μοντέλο μας. Όμως αν το size μικρύνει αρκετά, τότε λαμβάνουμε περισσότερο θόρυβο στο σύστημα μας και στις εκτιμήσεις των gradients.

- **Ερώτημα 7 :** Συνήθως ανακατεύουμε την σειρά των mini-batches στα δεδομένα εκπαίδευσης σε κάθε εποχή. Μπορείτε να εξηγήσετε γιατί?

Το shuffling των δεδομένων γίνεται για τρεις εξής δύο λόγους, μάλιστα καλή πρακτική θεωρείται να γίνεται shuffle στα train data και όχι στα test.

1. Επειδή το νευρωνικό έχει την δυνατότητα να μαθαίνει μη γραμμικότητες εξόδων συναρτήσει εισόδου, θέλουμε να αποφύγουμε και την προφανή του ικανότητα να μαθαίνει και την σειρά των δεδομένων που εισέρχονται στο νευρωνικό (γιατί δεν θα εκπαιδευτεί αποδοτικότερα), πράγμα το οποίο επιτυγχάνεται με shuffling των δεδομένων train σε κάθε εποχή. Με τον τρόπο αυτό δεν θα μάθει πάνω στην σειρά τους και ούτε θα την λαμβάνει υπόψιν του κατά την εκπαίδευση του.
2. Η σύγκλιση στο ολικό ελάχιστο είναι ο απώτερος σκοπός του stochastic gradient descent, επομένως οποιαδήποτε διαδικασία επιταχύνει την σύγκλιση είναι καλό να χρησιμοποιείται. Αντίστοιχα το shuffling παίζει πρωτεύοντα ρόλο στην κατεύθυνση αυτή, καθώς εάν στο νευρωνικό εισέρχονται συνεχώς διαφορετικά δεδομένα τότε το να ξεφύγουμε από ένα τοπικό ελάχιστο γίνεται με αμεσότητα σε επόμενες επαναλήψεις.

3.2 Βελτιστοποίηση - main.py

Για την βελτιστοποίηση του μοντέλου μας κάνουμε τις παρακάτω επιλογές-παραδοχές

- **Κριτήριο:** Και για δύο προβλήματα της κατηγοριοποίησης **Semeval2017A**, **MR** που αποτελούνται από 3 και 2 κλάσεις αντίστοιχα, χρησιμοποιείται το `CrossEntropyLoss()` ως κριτήριο-loss function
- **Παράμετροι:** Γίνεται επιλογή μόνο εκείνων που πράγματι θα βελτιστοποιηθούν, όσες δηλαδή έχουν `requires_grad = True` και επιτρέπουν τον υπολογισμό του gradient στο backpropagation
- **Optimizer:** Γίνεται επιλογή του αλγορίθμου βελτιστοποίησης Adam, για το οποίο ορίσαμε

Σημειώνεται ότι από την στιγμή που το μοντέλο μας δεν απαιτεί μέρες για την εκπαίδευση του, η επιλογή optimizer και των υπόλοιπων κριτηρίων δεν είναι άκρως σημαντική, καθώς μπορούμε πάντα να κάνουμε δοκιμές εναλλακτικής παραμετροποίησης συνεχώς. Ενδεικτικά έγιναν οι παραπάνω παραδοχές.

3.3 Εκπαίδευση - training.py

Υλοποιήσαμε τις μεθόδους `train_dataset()` και `eval_dataset()`, για την εκπαίδευση και την αξιολόγηση αντίστοιχα κάθε mini-batch.

- *train_dataset()*: Καλείται για κάθε batch σε μία εποχή του train set, δίνει τα δεδομένα εκπαίδευσης στο μοντέλο, υπολογίζει τις απώλειες-σφάλμα και τέλος ενημερώνει κατάλληλα τα βάρη του δικτύου με εφαρμογή του αλγορίθμου backpropagation
- *eval_dataset()*: Καλείται στο τέλος κάθε εποχής για να γίνει αξιολόγηση του μοντέλου, τόσο στο ίδιο το train set για στατιστικούς λόγους αλλά κυρίως στο test set για να εξετάσουμε την ακρίβεια και την αποτελεσματικότητά του.

Σημειώνεται ότι:

- Η κλήση του *model()* με input το mini batch και τον αντίστοιχο πίνακα πραγματικών μηκών αυτού, οδηγεί στην κλήση της συνάρτησης *forward()* την οποία ορίσαμε σε προηγούμενο βήμα, και η οποία επιστρέφει τις κλάσεις ως πρόβλεψη του μοντέλου.
- Το training του μοντέλου μας σταματάει με το που κάποιο cumulative running loss σε κάποια εποχή είναι μεγαλύτερο από το αντίστοιχο της προηγούμενης εποχής κατά έναν παράγοντα e.s

3.4 Αξιολόγηση - main.py

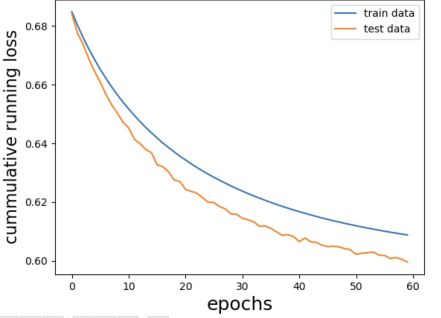
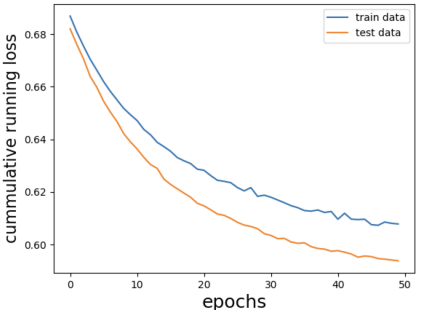
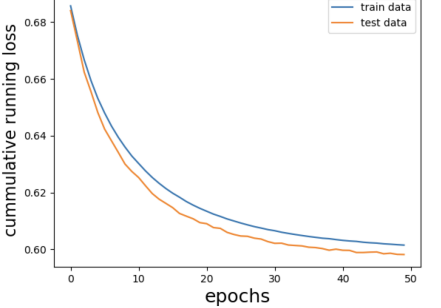
Το running του συνολικού κώδικα γίνεται με την παρακάτω διαδικασία

1. Ορίζουμε στο αρχείο **main.py** το προς εξέταση αρχείο, δηλαδή **Semeval2017A** ή **MR**, για τα οποία ο κώδικας εφαρμόζεται ορθά και δίνει διαφορετικά αποτελέσματα, τα οποία με αρκετές τροποποιήσεις και δοκιμές μπορούν να βελτιωθούν περαιτέρω, και τα οποία θα παρουσιάσουμε παρακάτω.
2. Ορίζουμε ποιο word embedding θα χρησιμοποιήσουμε από αυτά που υπάρχουν διαθέσιμα στον φάκελο */embeddings*
3. Γίνεται run στο command line ως εξής: *python3 main.py* ³. Στο stdout, φαίνονται τόσο τα αποτελέσματα των ερωτημάτων 1.1-1.3, όσο και training του μοντέλου σε κάθε εποχή με κατάλληλη κλήση της συνάρτησης *progress()* του **training.py**, όπως επίσης και οι ζητούμενες μετρικές του μοντέλου.
4. Τέλος, προβάλλουμε δύο γραφικές παραστάσεις κάνοντας κατάλληλη χρήση της συνάρτησης *plot()*, που αφορούν το cumulative running loss τόσο για train set όσο και για το data set.

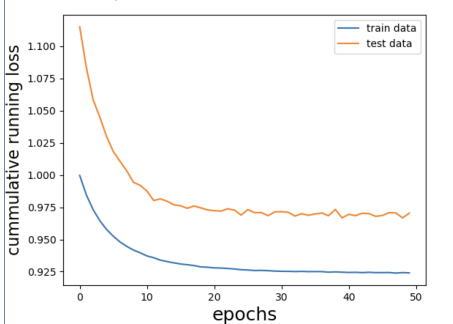
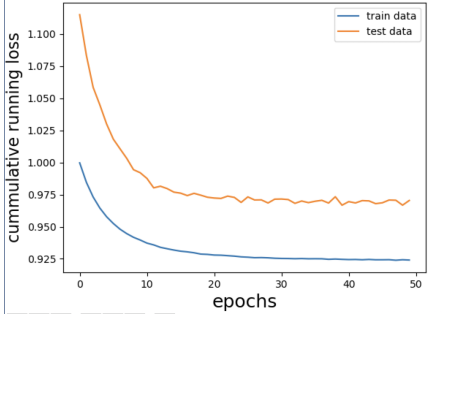
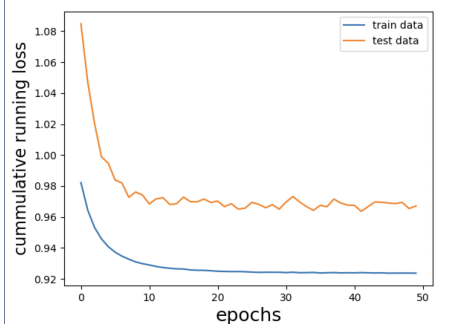
Οι διάφορες μετρικές υπολογίστηκαν κάνοντας κατάλληλη χρήση των αντίστοιχων συναρτήσεων της βιβλιοθήκης *sklearn*

³είτε βρισκόμαστε στο conda είτε όχι, απλά προσοχή στις βιβλιοθήκες μπορούν να χρησιμοποιηθούν σε κάθε περίπτωση και φαίνονται μέσω του python path

Για το αρχείο MR - 6B 50D embeddings

<p>EPOCHS = 60, Batch Size = 200,</p>	<p>Loss - epochs for both train and test set</p> 	<pre> F1_train: 0.6707362604349517 Accuracy_train: 0.6709 Recall_train: 0.6709 F1_test: 0.6580828579786456 Accuracy_test: 0.6586102719033232 Recall_test: 0.6586102719033232 </pre>	<p>-</p>
<p>EPOCHS = 50, Batch Size = 128,</p>	<p>Loss - epochs for both train and test set</p> 	<pre> F1_train: 0.6729377142583033 Accuracy_train: 0.673 Recall_train: 0.673 F1_test: 0.6610250080007315 Accuracy_test: 0.6616314199395771 Recall_test: 0.661631419939577 </pre>	<p>Το default size εποχών είναι πολύ μεγάλο εφόσον το loss του test set αρχίζει να αυξάνεται μετά από κάποια εποχή (για την ακρίβεια υπάρχουν διακυμάνσεις) το οποίο οφείλεται στο overfitting του μοντέλου στα train data. Απο αυτό καταλαβαίνουμε ότι η εκπαίδευση πρέπει</p>
<p>EPOCHS = 50, Batch Size = 50,</p>	<p>Loss - epochs for both train and test set</p> 	<pre> F1_train: 0.6749738606055624 Accuracy_train: 0.6751 Recall_train: 0.6751 F1_test: 0.668784483054042 Accuracy_test: 0.6691842900302115 Recall_test: 0.6691842900302115 </pre>	<p>-</p>

Για το αρχείο Semeval2017A - 6B 50D embeddings

<p>EPOCHS = 60, Batch Size = 200,</p>	<p>Loss - epochs for both train and test set</p> 	<p>F1_train: 0.4329232875942935 Accuracy_train: 0.5358591054893179 Recall_train: 0.4401020051497612 F1_test: 0.4763896294317749 Accuracy_test: 0.5188471871692584 Recall_test: 0.4868242582696076</p>	<p>-</p>
<p>EPOCHS = 50, Batch Size = 128,</p>	<p>Loss - epochs for both train and test set</p> 	<p>F1_train: 0.4329232875942935 Accuracy_train: 0.5358591054893179 Recall_train: 0.4401020051497612 F1_test: 0.4763896294317749 Accuracy_test: 0.5188471871692584 Recall_test: 0.4868242582696076</p>	<p>Το default size εποχών είναι πολύ μεγάλο εφόσον το loss του test set αρχίζει να αυξάνεται μετά από κάποια εποχή (για την ακρίβεια υπάρχουν διακυμάνσεις) το οποίο οφείλεται στο overfitting του μοντέλου στα train data. Απο αυτό καταλαβαίνουμε ότι η εκπαίδευση πρέπει να σταματήσει νωρίτερα, δεν υπάρχει περισσότερη βελτίωση</p>
<p>EPOCHS = 50, Batch Size = 50,</p>	<p>Loss - epochs for both train and test set</p> 	<p>F1_train: 0.4438282818156778 Accuracy_train: 0.5353749319130908 Recall_train: 0.4460778883504964 F1_test: 0.4877310236528376 Accuracy_test: 0.521533827240902 Recall_test: 0.49482626398693536</p>	<p>-</p>

Συμπεράσματα

Γενικότερα παρατηρούμε ότι όσο καλύτερα embeddings βάζουμε τόσο καλύτερα ήταν τα αποτελέσματα του μοντέλου μας, με το μειονέκτημα ότι γινόταν overfit με πολύ γρήγορο ρυθμό. Επίσης όσο μειωνόταν το batch size σε λογικά πλαίσια τόσο καλύτερα αποτελέσματα είχαμε σε επίσης, πολύ γρηγορότερος ο ρυθμός πτώσης του loss-σφάλματος.