Επεξεργασία Φωνής και Φυσικής Γλώσσας

Προπαρασκευή 3ου Εργαστηρίου: Επεξεργασία και κατηγοριοποίηση με χρήση βαθιών νευρωνικών δικτύων με χρήση της βιβλιοθήκης PyTorch

ΣΧΟΛΗ: ΗΜΜΥ

Ονοματεπώνυμο	Αριθμός Μητρώου
Γιάννης Πιτόσκας	03115077
Αντώνης Παπαοικονόμου	03115140

Σημείωση: Στην παρακάτω αναφορά έχει γίνει χρήση αρχείου glove.6B.50d.txt με τα pretrained embeddings.

1. Προεπεξεργασία Δεδομένων

Σε αυτό το σημείο καλούμαστε να επεξεργαστούμε τα δεδομένα, ώστε να μπορούμε να εκπαιδεύσουμε στη συνέχεια το νευρωνικό μας δίκτυο.

1.1 Κωδικοποίηση Επισημειώσεων (Labels)

Παρακάτω φαίνονται τα πρώτα 10 labels από τα δεδομένα εκπαίδευσης και οι αντιστοιχίες τους σε αριθμούς:

1.2 Λεκτική Ανάλυση (Tokenization)

Για το tokenization έχουμε χρησιμοποιήσει την συνάρτηση `word_tokenize()` του πακέτου `nltk`. Στη συνέχεια φαίνονται τα πρώτα 10 παραδείγματα από τα δεδομένα εκπαίδευσης:

1.3 Κωδικοποίηση Παραδειγμάτων (Λέξεων)

Κατόπιν υλοποίησης της μεθόδου `__getitem__ της κλάσης SentenceDataset`, τυπώνουμε 5 random παραδείγματα στην αρχική tokenized μορφή τους και ύστερα όπως τα επιστρέφει η κλάση `SentenceDataset`, δηλαδή με τα στοιχεία `example, label, length`. Σημειώνουμε επίσης πως για το μέγεθος του example array έχουμε χρησιμοποιήσει τον μέσο όρο των «μηκών» των προτάσεων που βρίσκονται στο εκάστοτε train set (21 για το 50d).

```
loading word embeddings...
Loaded word embeddings from cache.

EX1: First 10 train labels with encodings:

positive -> 1
positive -> 1
positive -> 1
positive -> 1
positive -> 1
positive -> 1
positive -> 1
positive -> 1
positive -> 1
positive -> 1
```

EX2: First 10 tokenized train data:

```
['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', '.']
['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', '.']
['effective', 'but', 'too-tepid', 'biopic']
['if', 'you', 'sometimes', 'like', 'to', 'go', 'to', 'the', 'movies', 'to', 'have', 'fun', ',', 'wasabi',
'is', 'a', 'good', 'place', 'to', 'start', '.']
['emerges', 'as', 'something', 'rare', ',', 'an', 'issue', 'movie', 'that', "'s", 'so', 'honest', 'and',
'keenly', 'observed', 'that', 'it', 'does', "n't", 'feel', 'like', 'one', '.']
['the', 'film', 'provides', 'some', 'great', 'insight', 'into', 'the', 'neurotic', 'mindset', 'of', 'all',
'comics', '--', 'even', 'those', 'who', 'have', 'reached', 'the', 'absolute', 'top', 'of', 'the', 'game',
'.']
['offers', 'that', 'rare', 'combination', 'of', 'entertainment', 'and', 'education', '.']
['perhaps', 'no', 'picture', 'ever', 'made', 'has', 'more', 'literally', 'showed', 'that', 'the', 'road',
'to', 'hell', 'is', 'paved', 'with', 'good', 'intentions', '.']
['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',
'.']
['take', 'care', 'of', 'my', 'cat', 'offers', 'a', 'refreshingly', 'different', 'slice', 'of', 'asian',
'cinema', '.']
```

EX3: 5 random SentenceDatasets from train set:

```
['a', 'trashy', ',', 'exploitative', ',', 'thoroughly', 'unpleasant', 'experience', '.']
[    8 53333     2 46593     2  9094 16764  1222     3     0     0     0
     0     0     0     0     0     0     0     0     0]
0
9
['comes', 'off', 'like', 'a', 'bad', 'imitation', 'of', 'the', 'bard', '.']
[  935   139   118     8   979 20300     4     1 23849     3     0     0
     0     0     0     0     0     0     0     0     0]
0
10
['.', '.', '.', 'a', 'haunting', 'vision', ',', 'with', 'images', 'that', 'seem', 'more', 'like',
'disturbing', 'hallucinations', '.']
[    3     3     3     8 18256  3139     2    18  3064    13  1915    57
   118  9393 36293     3     0     0     0     0     0]
1
16
['a', 'coming-of-age', 'film', 'that', 'avoids', 'the', 'cartoonish', 'clichés', 'and', 'sneering', 'humor',
'of', 'the', 'genre', 'as', 'it', 'provides', 'a', 'fresh', 'view', 'of', 'an', 'old', 'type', '--', 'the',
'uncertain', 'girl', 'on', 'the', 'brink', 'of', 'womanhood', '.']
[    8 82694   320    13 18246     1 45399 72456     6 70063  6203     4
     1  6617    20    21  1953     8  1904  1140     4]
1
34
['a', 'dumb', 'movie', 'with', 'dumb', 'characters', 'doing', 'dumb', 'things', 'and', 'you', 'have', 'to',
'be', 'really', 'dumb', 'not', 'to', 'see', 'where', 'this', 'is', 'going', '.']
[    8 14974  1006    18 14974  2154   915 14974   655     6    82    34
     5    31   589 14974    37     5   254   112    38]
0
24
```

2. Μοντέλο

Σε αυτό το βήμα γίνεται ο σχεδιασμός του νευρωνικού μας δικτύου.

2.1 Embedding Layer

- Γιατί αρχικοποιούμε το embedding layer με τα προεκπαιδευμένα word embeddings;

  Διότι άμα δεν αρχικοποιήσουμε το embedding layer με τα προεκπαιδευμένα word embeddings τότε τα weights παίρνουν random τιμές το οποίο φαίνεται να έχει χαμηλότερες επιδόσεις. Χρησιμοποιώντας την προσέγγιση των προεκπαιδευμένων word embeddings μπορούμε να τα χρησιμοποιήσουμε και να τα συντονίσουμε ιδανικά για τα δεδομένα.

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

  Δίοτι βλέπουμε ότι το embedding layer επωφελείται από αυτά τα προεκπαιδευμένα word embeddings. Τα προεκπαιδευμένα αυτά μέρη δε θα έπρεπε να ενημερώνονται κατά τη διάρκεια της εκπαίδευσης, για να αποφευχθεί το να «ξεχνιούνται» αυτά που ήδη «γνωρίζουν». Τα μεγάλα gradient updates που ενεργοποιούνται από layers που είναι τυχαία αρχικοποιημένα μπορούν να διαταράξουν τα ήδη γνωστά χαρακτηριστικά.

2.2 Output Layer(s)

Για την κατηγοριοποίηση θα πρέπει τώρα να προβάλλουμε τις αναπαραστάσεις των κειμένων στον χώρο των κλάσεων.

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

  Ο σκοπός της συνάρτησης ενεργοποίησης είναι η εισαγωγή μη γραμμικότητας στο δίκτυο. Μη γραμμικότητα σημαίνει κιόλας ότι η έξοδος δεν μπορεί να αναπαραχθεί από έναν γραμμικό συνδυασμό των εισόδων. Πρέπει να εφαρμόσουμε μια συνάρτηση ενεργοποίησης τέτοια ώστε το δίκτυο να γίνει πιο ισχυρό και να προσθέσει την ικανότητα να μάθει κάτι πολύπλοκο από τα δεδομένα και να αντιπροσωπεύει μη γραμμικές σύνθετες αυθαίρετες αντιστοιχίες μεταξύ εισόδου και εξόδου. Ως εκ τούτου, χρησιμοποιώντας μη γραμμική ενεργοποίηση, είμαστε σε θέση να παράγουμε μη γραμμικές απεικονίσεις από την είσοδο στην έξοδο.

  Στην περίπτωση, τώρα, γραμμικής συνάρτησης θα είχαμε σταθερό gradient ανεξάρτητο της εισόδου και επομένως το descent θα γίνει σε σταθερό gradient. Αν υπάρχει λοιπόν κάποιο σφάλμα στην πρόβλεψη, οι αλλαγές που έγιναν με το back propagation είναι σταθερές. Τώρα όσον αφορά τα connected layers. Κάθε layer ενεργοποιείται από γραμμική συνάρτηση. Αυτή η ενεργοποίηση με τη σειρά της πηγαίνει στο επόμενο layer ως είσοδος και το δεύτερο layer υπολογίζει το σταθμισμένο άθροισμα σε αυτή την είσοδο και με τη σειρά της, πυροδοτεί με βάση μια άλλη γραμμική συνάρτηση ενεργοποίησης. Ανεξάρτητα από το layers έχουμε, αν όλα είναι γραμμικά, η τελική συνάρτηση ενεργοποίησης του τελευταίου στρώματος δεν είναι τίποτε άλλο παρά μια γραμμική συνάρτηση της εισόδου του πρώτου layer. Αυτό σημαίνει ότι αυτά τα δύο layers (ή N layers) μπορούν να αντικατασταθούν από ένα μόνο layer. Όλο το δίκτυο λοιπόν εξακολουθεί να είναι ισοδύναμο με ένα μόνο layer με γραμμική συνάρτηση ενεργοποίησης (ένας συνδυασμός γραμμικών συναρτήσεων σε γραμμικό τρόπο είναι ακόμα μία γραμμική συνάρτηση).

2.3 Forward pass

Σε αυτό το σημείο θα σχεδιάστηκε ο τρόπος με τον οποίο θα μετασχηματίσει το δίκτυο τα δεδομένα εισόδου στις αντίστοιχες εξόδους.

- Αν θεωρήσουμε ότι 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);

  Διαισθητικά είναι σαν να κρατάμε με το κέντρο βάρους την «επικρατέστερη» τιμή για να μπορέσει να γίνει αντιληπτό σε ποια κατηγορία ανήκει ένα τμήμα δεδομένων (πχ θεματολογία).

- Αναφέρατε πιθανές αδυναμίες της συγκεκριμένης προσέγγισης για να αναπαραστήσουμε κείμενα.

  Στην περίπτωση των κειμένων μια αδυναμία αποτελεί το γεγονός ότι υπάρχουν ίδιες λέξεις με διαφορετική σημασιολογία. Ακόμη, σε μια πρόταση μπορεί να υπάρχουν λέξεις που κάθε μία απο αυτές εμφανίζεται συνηθέστερα σε διαφορετικές θεματολογίες. Γενικότερα, μπορεί η αναπαράσταση με το κέντρο βάρους να εμφανίσει αδυναμίες ορισμένες φορές, καθώς δεν είναι πάντα ενδεικτική μια τέτοιου είδους επιλογή σε κάποια ειδικά context.

3 Διαδικασία Εκπαίδευσης

3.1 Φόρτωση Παραδειγμάτων (DataLoaders)

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

  Τα δύο βασικά πράγματα που πρέπει να ληφθούν υπόψη κατά τη βελτιστοποίηση του μεγέθους mini-batches είναι η χρονική πολυπλοκότητα της εκπαίδευσης και η θορυβώδης εκτίμηση του gradient. Συγκεκριμένα, ο υπολογισμός του gradient είναι κατά προσέγγιση γραμμικός στο μέγεθος του batch. Επομένως, θα χρειαστεί περίπου 100 φορές περισσότερο για να υπολογίσετε το gradient ενός batch μεγέθους 10.000 από ένα μεγέθους 100. Επιπλέον, το gradient ενός μοναδικού σημείου δεδομένων θα είναι πολύ πιο θορυβώδες από το gradient ενός batch μεγέθους 100. Ακόμη, τα μεγάλα batch ρίχνουν την ποιότητα του μοντέλου, με κριτήριο την ικανότητα γενίκευσης, καθώς αυτή μειώνεται σε πολύ μεγάλο βαθμό. Στην πράξη χρησιμοποιούνται μικρού προς μεσαίου μεγέθους mini-batches περίπου 10-500. (εμείς επιλέξαμε 100 βασιζόμενοι και στα συμπεράσματα του επόμενου paper) (βλ. ON LARGE-BATCH TRAINING FOR DEEP LEARNING: GENERALIZATION GAP AND SHARP MINIMA, Published as a conference paper at ICLR 2017)
- Συνήθως ανακατεύουμε την σειρά των mini-batches στα δεδομένα εκπαίδευσης σε κάθε εποχή. Μπορείτε να εξηγήσετε γιατί;

  Εάν η σειρά των δεδομένων σε κάθε εποχή είναι ίδια, τότε το μοντέλο μπορεί να το χρησιμοποιήσει ως τρόπο μείωσης του σφάλματος του training, κάτι που είναι ένα είδος overfitting.

3.2 Βελτιστοποίηση

- Κριτήριο: Χρησιμοποιήθηκε η `CrossEntropyLoss()` μιας και εκτελεί εσωτερικά `softmax()`

- Παράμετροι: Βελτιστοποίηση των παραμέτρων για τις οποίες `p.requires_grad==True`

- Optimizers: Έγινε χρήση του `RMSProp()`

3.3 Εκπαίδευση

Για την αξιολόγηση κάθε batch ανάλογα με τα epochs που έχουμε επιλέξει (50 στην δική μας περίπτωση) γίνεται χρήση των συναρτήσεων `train_dataset()` και `eval_dataset()`.

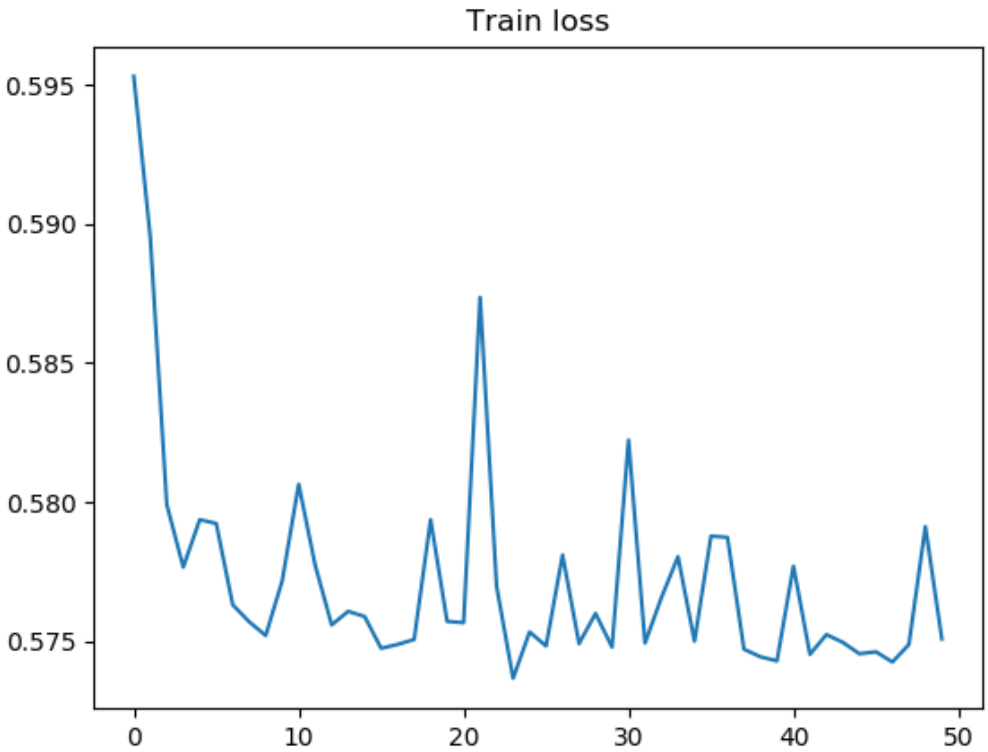
3.4 Αξιολόγηση

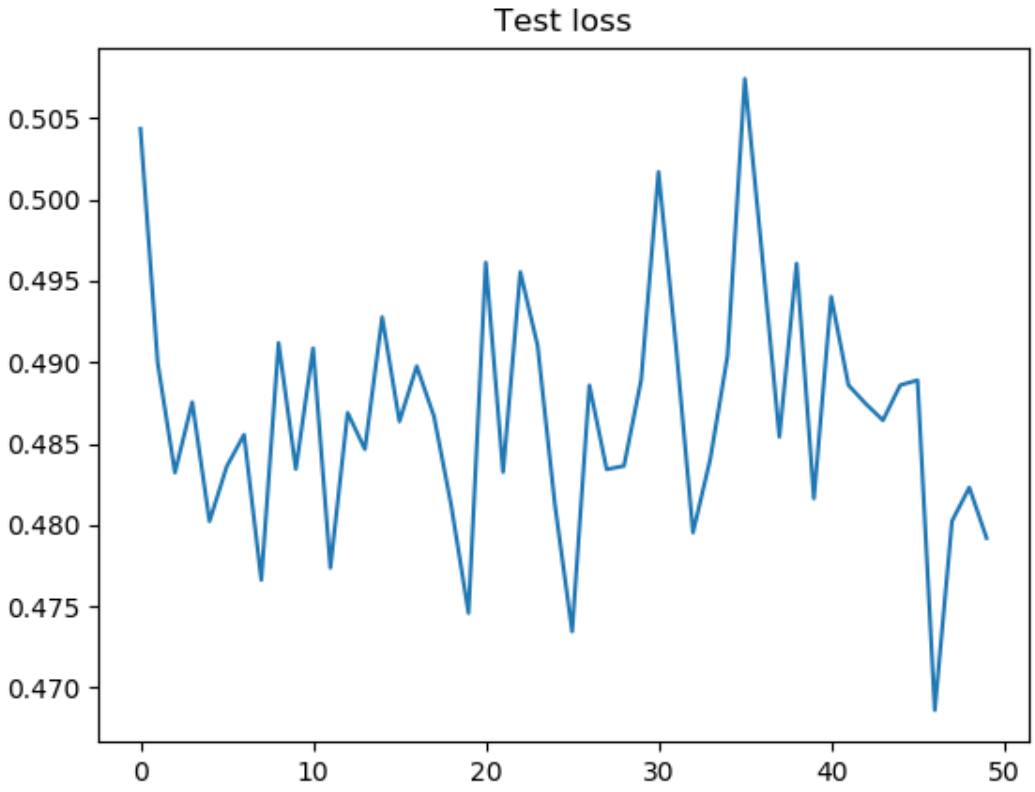
Για την αξιολόγηση του μοντέλου χρησιμοποιούμε τις μετρικές `accuracy`, `F1 score` (macro average) και `recall` (macro average). Για 50 εποχές λοιπόν στο testset MR και για το αρχείο `glove.6B.50d.txt` των embeddings έχουμε:

```
BaselineDNN(
  (embed): Embedding(400002, 50)
  (tanh): Tanh()
  (final): Linear(in_features=50, out_features=2, bias=True)
)
 [======================================-] ...Epoch 1, Loss: 0.5665
Train Set: loss=0.5924, accuracy=0.6782, f1=0.6782, recall=0.6782
Test Set: loss=0.4971, accuracy=0.6780, f1=0.6778, recall=0.6782
 [======================================-] ...Epoch 2, Loss: 0.5905
Train Set: loss=0.5836, accuracy=0.6846, f1=0.6841, recall=0.6857
Test Set: loss=0.4937, accuracy=0.6840, f1=0.6839, recall=0.6857
 [======================================-] ...Epoch 3, Loss: 0.5479
Train Set: loss=0.5801, accuracy=0.6867, f1=0.6865, recall=0.6873
Test Set: loss=0.4919, accuracy=0.6820, f1=0.6820, recall=0.6873
 [======================================-] ...Epoch 4, Loss: 0.5807
Train Set: loss=0.5798, accuracy=0.6861, f1=0.6840, recall=0.6912
Test Set: loss=0.4872, accuracy=0.6800, f1=0.6777, recall=0.6912
 [======================================-] ...Epoch 5, Loss: 0.5629
Train Set: loss=0.5799, accuracy=0.6863, f1=0.6841, recall=0.6919
Test Set: loss=0.4906, accuracy=0.7120, f1=0.7075, recall=0.6919
 [======================================-] ...Epoch 6, Loss: 0.6435
Train Set: loss=0.5771, accuracy=0.6906, f1=0.6899, recall=0.6924
Test Set: loss=0.4865, accuracy=0.6860, f1=0.6827, recall=0.6924
 [======================================-] ...Epoch 7, Loss: 0.5850
Train Set: loss=0.5771, accuracy=0.6902, f1=0.6898, recall=0.6910
Test Set: loss=0.5000, accuracy=0.6640, f1=0.6629, recall=0.6910
 [======================================-] ...Epoch 8, Loss: 0.4796
Train Set: loss=0.5760, accuracy=0.6880, f1=0.6877, recall=0.6884
Test Set: loss=0.4743, accuracy=0.7020, f1=0.7019, recall=0.6884
 [======================================-] ...Epoch 9, Loss: 0.5786
Train Set: loss=0.5784, accuracy=0.6870, f1=0.6857, recall=0.6901
Test Set: loss=0.4895, accuracy=0.6940, f1=0.6923, recall=0.6901
 [======================================-] ...Epoch 10, Loss: 0.5859
Train Set: loss=0.5740, accuracy=0.6908, f1=0.6905, recall=0.6914
Test Set: loss=0.4913, accuracy=0.6700, f1=0.6696, recall=0.6914
 [======================================-] ...Epoch 11, Loss: 0.5801
Train Set: loss=0.5773, accuracy=0.6853, f1=0.6848, recall=0.6865
Test Set: loss=0.4860, accuracy=0.6880, f1=0.6880, recall=0.6865
 [======================================-] ...Epoch 12, Loss: 0.6100
Train Set: loss=0.5748, accuracy=0.6904, f1=0.6903, recall=0.6909
Test Set: loss=0.4750, accuracy=0.7120, f1=0.7116, recall=0.6909
 [======================================-] ...Epoch 13, Loss: 0.5538
Train Set: loss=0.5758, accuracy=0.6896, f1=0.6895, recall=0.6899
Test Set: loss=0.4909, accuracy=0.6780, f1=0.6766, recall=0.6899
 [======================================-] ...Epoch 14, Loss: 0.5935
Train Set: loss=0.5760, accuracy=0.6882, f1=0.6881, recall=0.6883
Test Set: loss=0.4871, accuracy=0.6900, f1=0.6900, recall=0.6883
 [======================================-] ...Epoch 15, Loss: 0.6118
Train Set: loss=0.5742, accuracy=0.6932, f1=0.6928, recall=0.6941
Test Set: loss=0.4907, accuracy=0.6780, f1=0.6777, recall=0.6941
 [======================================-] ...Epoch 16, Loss: 0.5507
Train Set: loss=0.5752, accuracy=0.6937, f1=0.6935, recall=0.6944
Test Set: loss=0.4820, accuracy=0.6840, f1=0.6824, recall=0.6944
 [======================================-] ...Epoch 17, Loss: 0.6108
```

```
Train Set: loss=0.5762, accuracy=0.6877, f1=0.6876, recall=0.6877
Test Set: loss=0.4763, accuracy=0.6960, f1=0.6958, recall=0.6877
 [======================================-] ...Epoch 18, Loss: 0.5916
Train Set: loss=0.5751, accuracy=0.6887, f1=0.6886, recall=0.6888
Test Set: loss=0.4935, accuracy=0.6700, f1=0.6698, recall=0.6888
 [======================================-] ...Epoch 19, Loss: 0.5513
Train Set: loss=0.5755, accuracy=0.6895, f1=0.6884, recall=0.6917
Test Set: loss=0.4774, accuracy=0.7060, f1=0.7031, recall=0.6917
 [======================================-] ...Epoch 20, Loss: 0.5519
Train Set: loss=0.5754, accuracy=0.6924, f1=0.6922, recall=0.6930
Test Set: loss=0.4893, accuracy=0.6800, f1=0.6791, recall=0.6930
 [======================================-] ...Epoch 21, Loss: 0.5667
Train Set: loss=0.5759, accuracy=0.6879, f1=0.6879, recall=0.6879
Test Set: loss=0.4873, accuracy=0.6800, f1=0.6793, recall=0.6879
 [======================================-] ...Epoch 22, Loss: 0.5940
Train Set: loss=0.5749, accuracy=0.6908, f1=0.6906, recall=0.6912
Test Set: loss=0.4908, accuracy=0.6820, f1=0.6814, recall=0.6912
 [======================================-] ...Epoch 23, Loss: 0.5876
Train Set: loss=0.5783, accuracy=0.6788, f1=0.6776, recall=0.6815
Test Set: loss=0.4740, accuracy=0.6920, f1=0.6915, recall=0.6815
 [======================================-] ...Epoch 24, Loss: 0.5425
Train Set: loss=0.5746, accuracy=0.6924, f1=0.6917, recall=0.6945
Test Set: loss=0.4930, accuracy=0.6860, f1=0.6841, recall=0.6945
 [======================================-] ...Epoch 25, Loss: 0.6162
Train Set: loss=0.5752, accuracy=0.6865, f1=0.6862, recall=0.6874
Test Set: loss=0.4893, accuracy=0.6820, f1=0.6820, recall=0.6874
 [======================================-] ...Epoch 26, Loss: 0.5789
Train Set: loss=0.5759, accuracy=0.6901, f1=0.6901, recall=0.6902
Test Set: loss=0.4838, accuracy=0.6980, f1=0.6977, recall=0.6902
 [======================================-] ...Epoch 27, Loss: 0.6268
Train Set: loss=0.5751, accuracy=0.6922, f1=0.6922, recall=0.6924
Test Set: loss=0.4887, accuracy=0.6900, f1=0.6881, recall=0.6924
 [======================================-] ...Epoch 28, Loss: 0.5736
Train Set: loss=0.5750, accuracy=0.6872, f1=0.6870, recall=0.6879
Test Set: loss=0.4787, accuracy=0.7040, f1=0.7040, recall=0.6879
 [======================================-] ...Epoch 29, Loss: 0.5356
Train Set: loss=0.5759, accuracy=0.6861, f1=0.6858, recall=0.6870
Test Set: loss=0.4759, accuracy=0.7020, f1=0.7019, recall=0.6870
 [======================================-] ...Epoch 30, Loss: 0.4834
Train Set: loss=0.5748, accuracy=0.6897, f1=0.6896, recall=0.6898
Test Set: loss=0.4793, accuracy=0.6820, f1=0.6811, recall=0.6898
 [======================================-] ...Epoch 31, Loss: 0.5848
Train Set: loss=0.5748, accuracy=0.6923, f1=0.6920, recall=0.6932
Test Set: loss=0.4808, accuracy=0.6980, f1=0.6970, recall=0.6932
 [======================================-] ...Epoch 32, Loss: 0.5928
Train Set: loss=0.5751, accuracy=0.6922, f1=0.6920, recall=0.6929
Test Set: loss=0.4815, accuracy=0.6960, f1=0.6947, recall=0.6929
 [======================================-] ...Epoch 33, Loss: 0.5933
Train Set: loss=0.5744, accuracy=0.6922, f1=0.6917, recall=0.6936
Test Set: loss=0.4789, accuracy=0.6880, f1=0.6860, recall=0.6936
 [======================================-] ...Epoch 34, Loss: 0.6325
Train Set: loss=0.5745, accuracy=0.6929, f1=0.6928, recall=0.6930
Test Set: loss=0.5035, accuracy=0.6620, f1=0.6619, recall=0.6930
 [======================================-] ...Epoch 35, Loss: 0.6158
Train Set: loss=0.5756, accuracy=0.6830, f1=0.6820, recall=0.6851
Test Set: loss=0.4966, accuracy=0.6640, f1=0.6638, recall=0.6851
 [======================================-] ...Epoch 36, Loss: 0.5345
Train Set: loss=0.5750, accuracy=0.6916, f1=0.6912, recall=0.6927
Test Set: loss=0.4947, accuracy=0.6860, f1=0.6854, recall=0.6927
 [======================================-] ...Epoch 37, Loss: 0.6406
Train Set: loss=0.5770, accuracy=0.6832, f1=0.6823, recall=0.6849
Test Set: loss=0.4827, accuracy=0.6780, f1=0.6778, recall=0.6849
 [======================================-] ...Epoch 38, Loss: 0.5881
Train Set: loss=0.5751, accuracy=0.6907, f1=0.6902, recall=0.6922
Test Set: loss=0.4957, accuracy=0.6760, f1=0.6732, recall=0.6922
 [======================================-] ...Epoch 39, Loss: 0.6242
Train Set: loss=0.5753, accuracy=0.6871, f1=0.6870, recall=0.6875
Test Set: loss=0.4910, accuracy=0.6780, f1=0.6778, recall=0.6875
 [======================================-] ...Epoch 40, Loss: 0.5550
Train Set: loss=0.5800, accuracy=0.6792, f1=0.6773, recall=0.6836
Test Set: loss=0.4835, accuracy=0.6880, f1=0.6874, recall=0.6836
```

```
 [======================================-] ...Epoch 41, Loss: 0.5842
Train Set: loss=0.5746, accuracy=0.6905, f1=0.6905, recall=0.6906
Test Set: loss=0.4996, accuracy=0.6740, f1=0.6738, recall=0.6906
 [======================================-] ...Epoch 42, Loss: 0.6421
Train Set: loss=0.5753, accuracy=0.6855, f1=0.6853, recall=0.6859
Test Set: loss=0.4868, accuracy=0.6980, f1=0.6978, recall=0.6859
 [======================================-] ...Epoch 43, Loss: 0.5564
Train Set: loss=0.5735, accuracy=0.6908, f1=0.6908, recall=0.6908
Test Set: loss=0.4693, accuracy=0.6900, f1=0.6898, recall=0.6908
 [======================================-] ...Epoch 44, Loss: 0.6313
Train Set: loss=0.5843, accuracy=0.6819, f1=0.6764, recall=0.6954
Test Set: loss=0.5051, accuracy=0.6800, f1=0.6736, recall=0.6954
 [======================================-] ...Epoch 45, Loss: 0.6651
Train Set: loss=0.5751, accuracy=0.6933, f1=0.6930, recall=0.6940
Test Set: loss=0.4896, accuracy=0.6860, f1=0.6849, recall=0.6940
 [======================================-] ...Epoch 46, Loss: 0.6177
Train Set: loss=0.5765, accuracy=0.6823, f1=0.6816, recall=0.6840
Test Set: loss=0.4843, accuracy=0.6800, f1=0.6793, recall=0.6840
 [======================================-] ...Epoch 47, Loss: 0.5710
Train Set: loss=0.5745, accuracy=0.6888, f1=0.6887, recall=0.6889
Test Set: loss=0.4897, accuracy=0.6700, f1=0.6698, recall=0.6889
 [======================================-] ...Epoch 48, Loss: 0.6604
Train Set: loss=0.5742, accuracy=0.6922, f1=0.6922, recall=0.6923
Test Set: loss=0.4853, accuracy=0.6880, f1=0.6872, recall=0.6923
 [======================================-] ...Epoch 49, Loss: 0.5228
Train Set: loss=0.5735, accuracy=0.6931, f1=0.6926, recall=0.6942
Test Set: loss=0.4780, accuracy=0.6880, f1=0.6866, recall=0.6942
 [======================================-] ...Epoch 50, Loss: 0.5939
Train Set: loss=0.5826, accuracy=0.6843, f1=0.6798, recall=0.6951
Test Set: loss=0.4910, accuracy=0.6940, f1=0.6873, recall=0.6951
```




Test loss
0.505
0.500
0.495
0.490
0.485
0.480
0.475
0.470
0
10
20
30
40
50