ΠΜΣ-07 Τεχνητή Νοημοσύνη και Εφαρμογές στο Διαδίκτυο των Πραγμάτων
Τμήμα Πληροφορικής και Τηλεματικής
Χαροκόπειο Πανεπιστήμιο

3η Εργασία

Έκδοση 2023-1.0

Ημερομηνία Παράδοσης: 27/6/2023

Διδάσκων: Χρήστος Δίου
Επικουρική Διδασκαλία: Βασίλης Γκολέμης

1 Εισαγωγή

Στην εργασία αυτή θα ασχοληθούμε με το αντικείμενο της Επεξεργασίας Φυσικής Γλώσσας (Natural Language Processing). Στόχος μας είναι η επίλυση κάποιων θεμελιωδών προβλημάτων NLP. Πιο συγκεκριμένα, θα χρησιμοποιήσουμε το σύνολο των πακέτων σε Python που προσφέρονται από την πλατφόρμα HuggingFace. Η πλατφόρμα HuggingFace προσφέρει ένα πλήρες σετ από προεκπαιδευμένα μοντέλα κι ένα καλά σχεδιασμένο API για την εύκολη χρήση τους.

1.1 Παραδοτέα

Θα πρέπει να παραδώσετε ένα αρχείο `<name>.zip`, όπου `<name>` το επίθετό σας. Το αρχείο θα περιέχει **είτε** (α) τον κώδικα της υλοποίησής σας (ένα αρχείο .py ή ένα αρχείο .ipynb) και ένα αρχείο PDF με την αναφορά σας **είτε** (β) ένα αρχείο .ipynb που θα περιέχει και τον κώδικα της υλοποίησής σας και την αναφορά σας διατυπωμένη σε κατάλληλα Markdown κελιά ανάμεσα στον κώδικα. **είτε** (γ) ένα link σε ένα Colab Notebook που θα περιέχει και τον κώδικα της υλοποίησής σας και την αναφορά σας διατυπωμένη σε κατάλληλα Markdown κελιά ανάμεσα στον κώδικα.

2 Χρήση pretrained μοντέλων

Ένα από τα σημαντικότερα πλεονεκτήματα των πακέτων της HuggingFace, είναι η πολύ πλούσια συλλογή από προεκπαιδευμένα μοντέλα τα οποία μπορούν, με πολύ μικρή προγραμματιστική προσπάθεια να χρησιμοποιηθούν για την επίλυση των βασικών προβλημάτων NLP. Σε όλες τις περιπτώσεις, το API που θα χρησιμοποιήσετε είναι παρόμοιο με το παρακάτω:

```
from transformers import pipeline
model = pipeline(task=<task name>, model=<model name>)
model(<input>)
```

Βασισμένοι/ες στο παραπάνω κομμάτι κώδικα, κατεβάστε ένα κατάλληλο μοντέλο για τα παρακάτω tasks:

- Text Classification: Χαρακτηρισμός κειμένου εισόδου με ένα label, από ένα σύνολο από προκαθορισμένα labels. Για παράδειγμα, χαρακτηρισμός του κειμένου εισόδου ως θετικό/αρνητικό.
- Zero-Shot classification: Χαρακτηρισμός κειμένου εισόδου με ένα label, όπου όμως το σετ των labels δίνεται από τον χρήστη. Για παραδειγμα, ο χρήστης δηλώνει ότι θέλει το κείμενο εισόδου να χαρακτηριστεί ως `urgent, encouraging, insulting`.
- Token Classification: Χαρακτηρισμός κάθε token της εισόδου με ένα label. Πιο συγκεκριμένα, ασχοληθείτε με το πρόβλημα του part-of-speech tagging (POS), οπου σε κάθε token αποδίδεται ένα κατάλληλο μέρος του λόγου (ρήμα, υποκείμενο, αντικείμενο).

- Question Answering: Απάντηση σε μια ερώτηση σε φυσική γλώσσα. Το πρόβλημα αυτό χωρίζεται σε δύο υποκατηγορίες: απάντηση στην ερώτηση (α) με context (β) χωρίς context. Με context, δίνουμε φυσικό κείμενο για την ερώτηση και φυσικό κείμενο για το context και το μοντέλο επιστρέφει το κομμάτι του context που απαντάει στην ερώτηση. Χωρίς context, η απάντηση συντίθεται από το μηδέν.
- Summarization: Περίληψη του κειμένου εισόδου. Χωρίζεται, επίσης σε δύο κατηγορίες, (α) extractive, όπου επιλέγει τις πιο σημαντικές προτάσεις από το αρχικό κείμενο και (β) extractive, όπου η απάντηση συντίθεται από το μηδέν
- Translation: Μετάφραση από την γλώσσα του κειμένου εισόδου σε μια άλλη γλώσσα.
- Language modeling: Αποτελεί το πιο γενικό task. Το μοντέλο λαμβάνει το κείμενο εισόδου και παράγει ένα κείμενο εξόδου. Ανάλογα με το κείμενο εισόδου, αυτό το task μπορεί να επικαλύψει και κάποια από τα παραπάνω. Για παράδειγμα, αν δοθεί ως είσοδος η ερώτηση 'χαρακτήρισε το κείμενο "<παράθεση κειμένου>" ως θετική ή αρνητική κριτική', θα μπορούσε να υποκαταστήσει το πρόβλημα του text classification.

Σε κάθε ένα από τα παραπάνω tasks, δημιουργήστε μόνοι/ες σας 3-4 κατάλληλα inputs και δοκιμάστε το μοντέλο για το κάθε input. Σχολιάστε όλα τα predictions. Μείνατε ευχαριστημένοι/ες από το αποτέλεσμα; Θα εμπιστευόσασταν αυτό το μοντέλο σε ένα production περιβάλλον;

3 Finetuning ενός προεκπαιδευμένου μοντέλου

Τα προεκπαιδευμένα μοντέλα είναι συνήθως προσανατολισμένα στο να λύνουν γενικού τύπου tasks. Για παράδειγμα, ένα προεκπαιδευμένο μοντέλο που χαρακτηρίζει το κείμενο εισόδου ως θετικό ή αρνητικό (text classification) έχει συνήθως εκπαιδευτεί σε ένα μεγάλο εύρος κειμένων. Όμως συχνά χρειάζεται μια προσαρμογή ώστε το μοντέλο να γίνει πιο αποδοτικό σε συγκεκριμένα προβλήματα. Για παράδειγμα, ο χαρακτηρισμός του κειμένου ως θετικό/αρνητικό σε μια πλατφόρμα κινηματογράφου διαφέρει από το ίδιο πρόβλημα σε ένα αθλητικό site, διότι οι χρήστες χρησιμοποιούν διαφορετικά την γλώσσα στις δύο περιπτώσεις. Σε αυτές τις περιπτώσεις, είναι συνηθισμένο να χρησιμοποιούμε finetuning σε ένα πιο μικρό dataset, ενδεικτικό του επιμέρους task, ώστε να επιτυγχάνουμε την καλύτερη επίδοση. Παρακάτω, θα πρέπει να εφαρμόσετε finetuning σε ένα προεκπαιδευμένο μοντέλο χρησιμοποιώντας το σύνολο των πακέτων της HuggingFace μαζί με το πακέτο Deep Learning με το οποίο είστε πιο εξοικειωμένοι (Keras, Tensorflow, Pytorch κλπ).

3.1 Επιλογή ενός Dataset

Ως πρώτο βήμα, θα πρέπει να επιλέξετε ένα Dataset για text classification από την συλλογή του Hugging Face. Το Dataset μπορεί να είναι είτε binary (πχ. θετική ή αρνητική κριτική) ή multiclass (πχ. 1-5 αστεράκια). Είστε ελεύθεροι/ες να επιλέξετε όποιο Dataset επιθυμείτε. Κάποιες προτεινόμενες επιλογές είναι:

- imdb: αξιολόγηση κριτικών ταινιών της βάσης δεδομένων imdb σε θετικές/αρνητικές
- rotten tomatoes: αξιολόγηση κριτικών ταινιών της βάσης δεδομένων rotten tomatoes σε θετικές/αρνητικές
- Glue COLA: Corpus of Linguistic Acceptability, περιέχει φράσεις από κείμενα αγγλικών με αξιολόγηση αν είναι γραμματικά σωστές ή όχι

Λάβετε υπόψιν σας ότι τα χαρακτηριστικά του Dataset, π.χ. αριθμός εγγραφών, μέγεθος κειμένου, έχει επιπτώσεις στον χρόνο εκπαίδευσης, στην απαραίτητη μνήμη RAM κλπ. Οπότε επιλέξτε ένα Dataset το οποίο είναι συμβατό με την υποδομή που έχετε διαθέσιμη.

- Χρησιμοποιείστε την συνάρτηση `load_dataset_builder(<dataset name>)` για να εξερευνήσετε τα χαρακτηριστικά του dataset
- Χρησιμοποιείστε την συνάρτηση `load_dataset(<dataset name>)` για να κατεβάσετε το επιθυμητό dataset και να το χωρίσετε σε train/validation/test set.

3.2 Preprocessing και Finetuning

Απαραίτητη προϋπόθεση για την χρήση ενός NLP dataset στο finetuning ενός μοντέλου είναι η μετατροπή του αρχικού κειμένου (raw text) σε μια κατάλληλη διανυσματική αναπαράσταση, μια διαδικασία γνωστή ως tokenization. Για τον σκοπό αυτό θα σας φανούν χρήσιμες οι παρακάτω συναρτήσεις:

```
from transformers import AutoTokenizer
tokenizer = AutoTokenizer.from_pretrained(<pretrained tokenizer name>)
tokenized_data = tokenizer(<args>)
```

Δώστε ιδιαίτερη προσοχή στην επιλογή κατάλληλων παραμέτρων στον μετασχηματισμό, δηλαδή ορίστε κατάλληλα τα ορίσματα `padding, truncation` κλπ, και δικαιολογήστε την απόφασή σας. Αφού ορίσετε έναν κατάλληλο tokenizer, εφαρμόστε τον στα δεδομένα σας. Συγκεκριμένα:

- Εφαρμόστε τον `tokenizer` σε ένα συγκεκριμένο instance του dataset και δείτε το αποτέλεσμα του. Σχολιάστε τι παρατηρείτε. Σε ποιές πληροφορίες (διανύσματα) μετατράπηκε το αρχικό text και ποια η σημασία του κάθε ενός;
- Εφαρμόστε τον `tokenizer` συστηματικά σε όλο το dataset

Για το finetuning, χρειάζεται να επιλέξετε το πακέτο Deep Learning που προτιμάτε (Keras, Pytorch) και έπειτα να υλοποιηθούν δυο βήματα:

- Μετατροπή του dataset από ένα αντικείμενο του πακέτου Dataset της Hugging Face, σε ένα αντικείμενο συμβατό με το πακέτο Deep Learning που θα χρησιμοποιήσετε. Μπορείτε να διαβάσετε τους οδηγούς για μετατροπή σε object συμβατό με το Tensorflow για χρήση με το Keras ή σε object συμβατό με το Pytorch.
- Κατέβασμα ενός pretrained μοντέλου συμβατό με το πακέτο Deep Learning που θα χρησιμοποιήσετε. Θα σας φανούν χρήσιμα τα πακέτα `transformers.TFAutoModelForSequenceClassification` για το Keras και `transformers.AutoModelForSequenceClassification` για το Pytorch.

Εφ' όσον έχετε ολοκληρώσει με επιτυχία τα παραπάνω βήματα, μπορείτε πλέον να εφαρμόσετε finetuning. Εκπαιδεύστε το μοντέλο

3.3 Finetuning

Χρησιμοποιήστε το training set του Dataset που επιλέξατε για να εφαρμόσετε finetuning. Μπορείτε να χρησιμοποιείστε τις παρακάτω παραμέτρους αλλά και πειραματιστείτε με άλλες τιμές αν το επιθυμείτε.

- Τον αλγόριθμο βελτιστοποίησης Adam με ρυθμό εκμάθησης 10^3 , $\beta_1 = 0.9$ (ρυθμός ενημέρωσης πρώτης ροπής) $\beta_2 = 0.99$ (ρυθμός ενημέρωσης δεύτερης ροπής)
- Την κατηγορική διεντροπία (binary/multiclass cross-entropy) ως συνάρτηση απώλειας (loss function)
- Την ορθότητα (accuracy) ως μετρική αξιολόγησης
- 10 εποχές μέγιστη διάρκεια εκπαίδευσης
- Πρόωρο τερματισμό της εκπαίδευσης (Early Stopping) αν δεν παρουσιαστεί μείωση της απώλειας στο σύνολο επικύρωσης για 5 συνεχείς εποχές

3.4 Αξιολόγηση

Για την αξιολόγηση του μοντέλου χρησιμοποιείστε το σύνολο επικύρωσης (validation set). Θα χρειαστεί να δημιουργήσετε μια συνάρτηση:

```
confusion_matrix(model)
```

που θα επιστρέφει τον πίνακα σύγχυσης (confusion matrix). Με βάση τον πίνακα σύγχυσης υπολογίστε (a) τις γενικές (σε όλο το dataset) τιμές accuracy, precision, recall, καθώς και (b) τις τιμές precision, recall, f1 score ανα κλάση. Σχολιάστε την επίδοση του μοντέλου σας.