Coding Project

Κυρίτσης Βασίλειος 02999 και Σταμούλος Αλέξανδρος 02954

Νευρο-Ασαφής Υπολογιστική 2023-24
Τμήμα Ηλεκτρολόγων Μηχανικών & Μηχανικών Υπολογιστών
Πανεπιστήμιο Θεσσαλίας, Βόλος
{vakyritsis, astamoulos}@e-ce.uth.gr

1 Δεδομενα

1.1 Data inspection

Στο dataset που μας δόθηκε, παρατηρούμε τις ακόλουθες στήλες: data_id, id, date, source, title, content, author, url, published, published_utc, collection_utc, category_level_1, category_level_2. Οι στήλες category_level_1 και category_level_2 αποτελούν το στόχο (target) των νευρωνικών μας δικτύων για το επίπεδο 1 και το επίπεδο 2 αντίστοιχα.

Κατά την επιλογή των χαρακτηριστικών (features), επιλέξαμε να χρησιμοποιήσουμε τις στήλες source, title, content και author. Παρατηρήσαμε ότι η στήλη author περιέχει απουσιάζουσες τιμές (NaN), και μετά από ανάλυση διαπιστώσαμε ότι περίπου ένα στα τρία entries δεν περιλαμβάνουν πληροφορίες σχετικά με τον συγγραφέα. Επιπλέον, κάθε συγγραφέας έχει συνήθως γράψει λίγα άρθρα, κάτω από 30, με πολλούς να έχουν γράψει λιγότερα από 10. Τέλος, παρατηρήσαμε πολλούς συγγραφείς να έχουν συνεισφέρει σε πολλές κατηγορίες. Με βάση αυτά τα ευρήματα, αποφασίσαμε να μην συμπεριλάβουμε τη στήλη author ως feature στο μοντέλο μας.

Ενώσαμε αυτες τις τρεις στήλες ('**source**', '**title**', '**content**') για να πάρουμε τα τελικά features μας. Σε ολα μας τα μοντελα χωρισαμε το dataset σε train, test και validation (70%, 15%, 15%)

1.2 Preprocess

Η διαδικασία προεπεξεργασίας κειμένου που ακολουθήσαμε βασίστηκε στην μέθοδο που περιγράφεται στην αναφορά [3] του paper [4]. Το preprocess περιλάμβανε τα εξής:

- αφαίρεση των links(https)
- αφαίρεση των ip adresess
- αφαίρεση bad symbols
- αφαίρεση διαδοχικών κενών χαρακτήρων

Στο πλαίσιο του paper [4], περιγράφονται επιπλέον βήματα προεπεξεργασίας, όπως η αφαίρεση των ''stopwords'' η αφαίρεση μικρών λέξεων και η τεχνική stemming. Ωστόσο, αποφασίσαμε να μην υλοποιήσουμε αυτά τα επιπλέον βήματα, οι λόγοι για αυτήν την απόφαση θα αναλυθούν παρακάτω, κατά την περιγραφή της αρχιτεκτονικής του μοντέλου.

2 Περιγραφή μοντέλου

Κατανοήσαμε ότι η απόδοση των classifiers μειώνεται όσο αυξάνεται ο αριθμός των κατηγοριών. Το dataset μας περιλαμβάνει 109 κατηγορίες δεύτερου επιπέδου, καθιστώντας δύσκολο το επίτευγμα υψηλής ακρίβειας (accuracy) με τα κλασικά μοντέλα. Συνεπώς, η γενική μας ιδέα είναι η εξής: Για το πρώτο επίπεδο, θα χρησιμοποιήσουμε ένα αναδραστικό νευρωνικό δίκτυο (RNN), το οποίο έχει τη δυνατότητα να ανιχνεύει μακροπρόθεσμες εξαρτήσεις και να χειρίζεται ακολουθιακά δεδομένα. Για το δεύτερο επίπεδο, σκοπεύουμε να χρησιμοποιήσουμε ένα stack από αρχιτεκτονικές βαθιάς μάθησης, όπου κάθε μία από αυτές θα παρέχει εξειδικευμένη κατανόηση για κάθε κατηγορία του πρώτου επιπέδου. Και για τις δυο κατηγορίες δοκιμάσαμε RNN και DNN μοντέλα, για το επίπεδο 1 επιλέξαμε το RNN και για το επίπεδο 2 το DNN.

2.1 Επίπεδο 1

Αρχικά, δημιουργήσαμε ένα TextVectorization layer στο TensorFlow, όπου περάσαμε τη συνάρτηση προεπεξεργασίας ως όρισμα. Έτσι, ενσωματώσαμε τη διαδικασία προεπεξεργασίας μέσα στο TextVectorization layer. Στη συνέχεια, δημιουργήσαμε ένα επίπεδο embedding layer χρησιμοποιώντας τα pre-trained embeddings GloVe (Global Vectors for Word Representation) [2]. Επιλέξαμε να επιτρέψουμε την ενημέρωση των βαρών του επιπέδου ενσωμάτωσης κατά τη διάρκεια της εκπαίδευσης (trainable=True), η χρήση των GloVe embeddings μας έδωσε αρκετά καλύτερα απόδοση σε σχέση με ένα απλό embedding layer.

Το κύριο επίπεδο του μοντέλου μας είναι ένα Bidirectional GRU με διάσταση 896 και παράμετρο dropout 0.35. Η χρήση του Bidirectional επιτρέπει την αξιοποίηση πληροφοριών τόσο από τις προηγούμενες όσο και από τις επόμενες λέξεις, ενισχύοντας έτσι την ικανότητα μοντέλου να κατανοεί τα συμφραζόμενα. Τέλος, χρησιμοποιήσαμε ένα Dense layer με διάσταση ίση με τον αριθμό των κατηγοριών επιπέδου και συνάρτηση ενεργοποίησης 'softmax'. Ως συνάρτηση απώλειας (loss function) επιλέξαμε την Sparse Categorical Crossentropy, ενώ ως βελτιστοποιητή χρησιμοποιήσαμε τον Adam. Η επιλογή της **softmax** είναι κατάλληλη για προβλήματα multi-class ταξινόμησης, ενώ η **Sparse Categorical Crossentropy** είναι η κατάλληλη επιλογή για πολυκατηγορική ταξινόμηση όταν οι ετικέτες είναι ακέραιοι (όπως 0, 1, 2, κλπ.).

Παρατηρησεις: Κατά τη διάρκεια της ανάπτυξης του μοντέλου, πραγματοποιήσαμε πολλές δοκιμές προκειμένου να βελτιστοποιήσουμε την απόδοση. Ανάμεσα στις παραμέτρους που δοκιμάσαμε συμπεριλαμβάνονται το μέγεθος του λεξιλογίου (vocabulary), οι optimizer όπως ο Adam, ο Rmsprop, και ο AdamW, το ποσοστό του dropout, καθώς επίσης και πιο "deep" μοντέλα με περισσότερα GRU layers. Οι δοκιμές έγιναν τόσο χειροκίνητα, προσαρμόζοντας τις παραμέτρους με βάση την εμπειρία και την κατανόησή μας, όσο

και με τη χρήση του Keras Tuner. Το Keras Tuner μας βοήθησε να εξερευνήσουμε αυτόματα τον χώρο των υπερπαραμέτρων και να βρούμε βέλτιστες τιμές για τις παραμέτρους του μοντέλου μας. Καταλήξαμε στις εξής βέλτιστες παραμέτρους για το μοντέλο μας: GRU layer dimension ίση με **896**, ποσοστό dropout ίσο με **0.35**, και ως optimzer ο **Adam**. χρησιμοποιήσαμε 5 epochs καθώς για παραπάνω παρατηρούσαμε overfitting , ενώ το loss κατέβαινε, το validation loss αυξανόταν.

Επίσης, αποφασίσαμε να μην αφαιρέσουμε τα stopwords και να μην κάνουμε κάποια επιπλέον βηματα του preprocess. Η ακολουθία των λέξεων έχει σημασία στο GRU, και η αφαίρεση των stopwords θα μπορούσε να οδηγήσει στην απώλεια σημαντικού context. Αυτό επιβεβαιώθηκε πειραματικά, ενισχύοντας τη σημασία της διατήρησης ορισμένων στοιχείων του προεπεξεργαστικού βήματος για την απόδοση του μοντέλου μας.

2.2 Επίπεδο 2

Η βασική ιδέα πίσω από την αρχιτεκτονική μας για το δεύτερο επίπεδο κατηγοριοποίησης βασίστηκε στο paper [1]. Στόχος μας ήταν να δημιουργήσουμε ένα stack από μοντέλα, όσα και τα επίπεδα κατηγοριών του πρώτου επιπέδου (συνολικά 17 κατηγορίες).

Αρχικά, χωρίσαμε το αρχικό dataset σε 17 υπο-datasets, κάθε ένα αφορούσε δεδομένα μόνο για μια κατηγορία πρώτου επιπέδου.΄Επειτα, δημιουργήσαμε 17 μοντέλα, όπου καθένα εκπαιδεύτηκε με τα αντίστοιχα δεδομένα από μια κατηγορία.

Για το δεύτερο επίπεδο, χρησιμοποιήσαμε ολοκληρη της διαδικασιας του preprocess [3], όπου αντί για stemming εφαρμόσαμε lemmatization, καθώς δεν σκοπεύαμε να χρησιμοποιήσουμε RNN. Για το DNN, χρησιμοποιήσαμε count-based και term frequency–inverse document frequency (tf-idf) για την εξαγωγή χαρακτηριστικών.

Αναλυτικά, χρησιμοποιήσαμε ένα TfidfVectorizer() με max_features 20000 και χρήση bi-grams. Στη συνέχεια, ενσωματώσαμε ένα Dense layer με διάσταση 400, ένα dropout layer με ποσοστό 0.5 για αποφυγή υπερ-προσαρμογής (overfitting), και τέλος ένα Dense layer με διάσταση ίση με τον αριθμό των κατηγοριών του επιπέδου 2 για τη συγκεκριμένη κατηγορία επιπέδου 1.

Για τους ίδιους λόγους με το πρώτο επίπεδο, χρησιμοποιήσαμε activation function την 'softmax' και loss function την SparseCategoricalCrossentropy(). Επίσης για optimizer επιλέξαμε τον rmsprop. Η διαδικασία αυτή επαναλαμβάνεται για κάθε κατηγορία πρώτου επιπέδου.

Παρατηρήσεις Το paper [1] για το DNN προτείνει 8 hidden layers με 1024 cells σε κάθε hidden layer. Ωστόσο, στα πειράματά μας, στα δικά μας πειράματα παρατηρήσαμε ότι το μοντέλο βελτιώνεται με τη μείωση του αριθμού των hidden layer. Τελικά καταλήξαμε σε ένα dense layer με 400 cells αντί για 1024 cells που προτείνει το άρθρο. Η διαδικασία του hyperparameter tuning για τα παραπάνω αλλά και για το max_features που χρησιμοποιείται στο tf-idf vectorizer έγινε όπως και στο επίπεδο ένα χειροκίνητα και με την χρήση

του keras tuner. Επίσης επειδή τώρα έχουμε 17 μοντέλα , είναι δύσκολο να επιλέξουμε μόνοι μας τον κατάλληλο αριθμό των epochs για την εκπαίδευση. Για την αυτοματοποίηση της διαδικασίας αυτής, χρησιμοποιήσαμε το EarlyStopping από το module keras.callbacks. Το EarlyStopping διακόπτει την εκπαίδευση όταν το validation loss σταματά να μειώνεται, επιλέγοντας των βέλτιστο αριθμό epochs για τη συγκεκριμένη κατηγορία.

Το summary του νευρωνικό του 1ου επίπεδου

Model: Level 1 Model

Layer (type)	Output Shape	Param #
embedding_13 (Embedding)	(None, None, 100)	800200
bidirectional_13 (Bidirectional)	(None, 1792)	5365248
dense_13 (Dense)	(None, 17)	30481

Total params: 6,195,929
Trainable params: 6,195,929
Non-trainable params: 0

Το summary από ένα νευρωνικό του 2ου επίπεδου

Model: Level 2 Model

Layer (type)	Output Shape	Param #
dense_34 (Dense)	(None, 400)	8000400
dropout_17 (Dropout)	(None, 400)	0
dense_35 (Dense)	(None, 3)	1203

Total params: 8,001,603
Trainable params: 8,001,603
Non-trainable params: 0

3 Αποτελέσματα

Για το πρώτο επίπεδου έχουμε

– accuracy: 0.8437 - loss: 0.6194
– train time: 404 seconds (6min and 44 sec)
– inference time: 7 seconds

Για το δεύτερο επίπεδο έχουμε

0.8888
0.8916
0.7333
0.7066
0.7833
0.5714
0.8555
0.7904
0.8222
0.6666
1.0
0.7037
0.625
0.6583
0.8848
0.8455
0.7166

Πίνακας 1. Accuaracy για κάθε νευρωνικό του level 2

– M.O accuaracy 0.7731
– train time: 73.4 seconds (1min and 13.4sec)
– inference time: 2.2 seconds

Τα τελικά parameters δίνονται σε ξεχωριστά αρχεία

References

[1] Kamran Kowsari et al. "HDLTex: Hierarchical Deep Learning for Text Classification". In: *2017 16th IEEE International Conference on Machine Learning and Applications (ICMLA)*. IEEE, Dec. 2017. DOI: 10.1109/icmla.2017.0-134. URL: http://dx.doi.org/10.1109/ICMLA.2017.0-134.

[2] Jeffrey Pennington, Richard Socher, and Christopher D. Manning. "GloVe: Global Vectors for Word Representation". In: *Empirical Methods in Natural Language Processing (EMNLP)*. 2014, pp. 1532–1543. URL: http://www.aclweb.org/anthology/D14-1162.

[3] Alina Petukhova. *mn-ds-news-classification: A News Classification Project*. https://github.com/alinapetukhova/mn-ds-news-classification. 2023.

[4] Alina Petukhova and Nuno Fachada. "MN-DS: A Multilabeled News Dataset for News Articles Hierarchical Classification". In: *Data* 8.5 (2023). ISSN: 2306-5729. DOI: 10.3390/data8050074. URL: https://www.mdpi.com/2306-5729/8/5/74.