Paraphrase - detekce vztahů vět v textu

Matej Berezný <xberez03@stud.fit.vutbr.cz>
Ondrej Valo <xvaloo00@stud.fit.vutbr.cz>
Miloš Uriga <xuriga00@stud.fit.vutbr.cz>

29. května 2022

1 Úvod

Cieľom projektu bolo vytvoriť program pre klasifikáciu duplikátov otázok. Ako nástroj pre daný účel sme sa rozhodli využiť transformer model BERT. Dodatočne bolo toto zadanie rozšírené o pridanie modifikácií pre danú architektúru a porovnanie s existujúcimi riešeniami.

2 Dataset

Ako dataset bol využitý Quora Question Pairs dataset [Kag07]. Jedná sa o klasifikačný problém o dvoch triedach kde buď dve otázky boli duplikáty alebo nie. Dáta obsahovali 6 stĺpcov uvedených v tabuľke 1 aj s príkladom obsahu.

id	qid1	qid2	question1	question2	is_duplicate
38	75	76	How do we prepare for UPSC?	How do I prepare for civil service?	1

Tabulka 1: Hlavička dátovej sady

- id - identifikačné číslo data
- qid1 - identifikačné číslo prvej otázky
- qid2 - identifikačné číslo druhej otázky
- question1 - prvá otázka na porovnanie
- question2 - druhá otázka na porovnanie
- is_duplicate - 1 pre duplikátne otázky 0 pre nie

3 Augmentácia

Augmentácia predstavená v štúdií [WZ19] popísaná v podsekciách 3.1, 3.2, 3.3 a v 3.4. Tento spôsob augmentácie sme zvolili preto, že autorom vyššie spomínanej štúdie priniesla výrazné zlepšenie v presnosti trénovaných sietí. Rozhodli sme sa preto teda otestovať, či aplikovanie takejto augmentácie zlepší aj výsledky hľadania duplicitných otázok, poprípade či bude mať aplikácia takejto augmentácie na 'cutting-edge' modely typu BERT pozitívny vplyv na ich efektivitu. Rovnako ako v [WZ19] bola augmentácia prevádzaná na vetách podla dĺžky viet kde počet augmentácií n bol počítaný nasledovne: $n = l \times p$ kde l predstavuje dĺžku vety a p predstavuje pravdepodobnosť aplikovania augmentácia. Taktiež sme v augmentácií využili parameter N ktorí určoval celkový počet augmentovaných viet na jednu pôvodnú vetu.

3.1 Zámena synoným (SR)

Augmentačná technika kde boli zamieňané náhodné slová vo vete za ich synonymá. Avšak tieto slová nemohli byť takzvané "stopwords". Čo je určitá skupina slov ktoré filtrujeme pre nezvyšovanie zbytočného šumu v rámci dát, medzi takéto v našom prípade patria napríklad 'i', 'me', 'my', 'myself', 'we', 'our' [Wik21]. Pre hľadanie synoným bol použitý natrénovaný model wordnet [NLT22]. Kde následne na základe vstupnej pravdepodobnosti sa augmentovali vety 1 krát alebo viac a pridali k ostatným augmentovaným vetám. Príklad takejto augmentácie možno vidno v 2.

pôvodná veta:	A sad, superior human comedy played out on the back roads of life
augmentovaná veta:	A **lamentable**, superior human comedy played out on the **backward** road of life.

Tabulka 2: Príklad zámeny synonymom

3.2 Náhodné vsunutie (RI)

Táto technika funguje podobným spôsobom ako 3.1, kde narozdiel nahradzovania slov za synonymá, sa tieto synonymá vkladajú do pôvodnej vety. Príklad takejto augmentácie možné pozorovať v tabuľke 3.

pôvodná veta:	A sad, superior human comedy played out on the back roads of life
augmentovaná veta:	A sad, superior human comedy played out on **funniness** the back roads of life.

Tabulka 3: Príklad vkladania synonyma do vety

3.3 Náhodná zámena (RS)

Tu augmentácia prebiehala spôsobom že z dát sa vyberú dve slová z danej vety a vymenia si svoje pozície. Táto technika sa taktiež vykoná 1 krát a viac podľa dĺžky vety a určenej pravdepodobnosti. Príklad takejto augmentácie je možné pozorovať v tabuľke 4.

pôvodná veta:	A sad, superior human comedy played out on the back roads of life
augmentovaná veta:	A sad, superior human comedy played out on **roads** back **the** of life

Tabulka 4: Príklad zámeny slov vo vete

3.4 Náhodné zmazanie (RD)

V tejto augmentačnej technike sa prechádzajú všetky slová a každé má pravdepodobnosť byť z vety odstránené. Pravdepodobnosť je ručne nastavovaná a rovnaká pre každé slovo v danej vete. Príklad takejto augmentácie je možné pozorovať v tabuľke 5.

pôvodná veta:	A sad, superior human comedy played out on the back roads of life
augmentovaná veta:	A sad, superior human out on the roads of life

Tabulka 5: Príklad odstránenia slov vo vete

4 Architektúra

Architektúry využité v tomto projekte sú opísané v nasledujúcivh podsekciách BERT v 4.1, LSTM v 4.2, CNN v 4.3 a LSTMCNN v 4.4. Dôvodom využitia všetkých modelov v siamese sieťach je z dôvodu povahy problému, kde naše dáta sa skladajú z 2 otázkou a 1 labelu. Kde vstupné sekvencie na uvedených modeloch 1, 2, 3, 4 boli páry otázok kde každá sekvencia bola jedna z otázka.

4.1 BERT

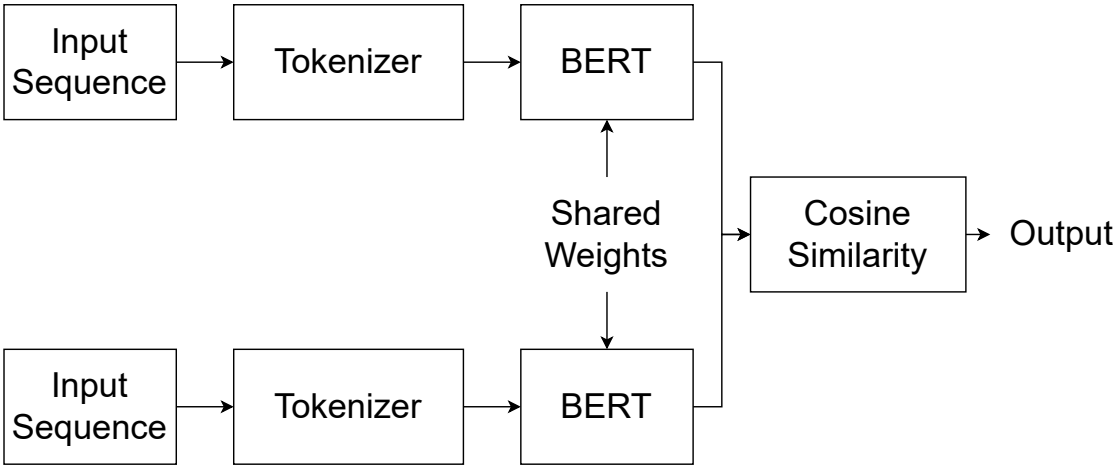
Na implementáciu BERTu bola využitá knižnica transformers [Fac22], kde pre siamese verziu sme museli preťažovať niektoré funkcie pre trénovanie. V obrázku 1, input sequence predstavujú otázky pre porovnávanie.

Tokenizer je taktiež využitý z knižnice transformers ktorý danú vetu zakóduje, jeho vstup a výstup je možno vidieť v tabulke 6.

Vstupná veta:	where is Himalayas in the world map?
Zakódovaná veta :	[101, 2073, 2003, 26779, 1999, 1996, 2088, 4949, 1029, 102]
kódovanie na tokeny :	['[CLS]', 'where', 'is', 'himalayas', 'in', 'the', 'world', 'map', '?', '[SEP]']

Tabulka 6: Príklad fungovania tokenizeru

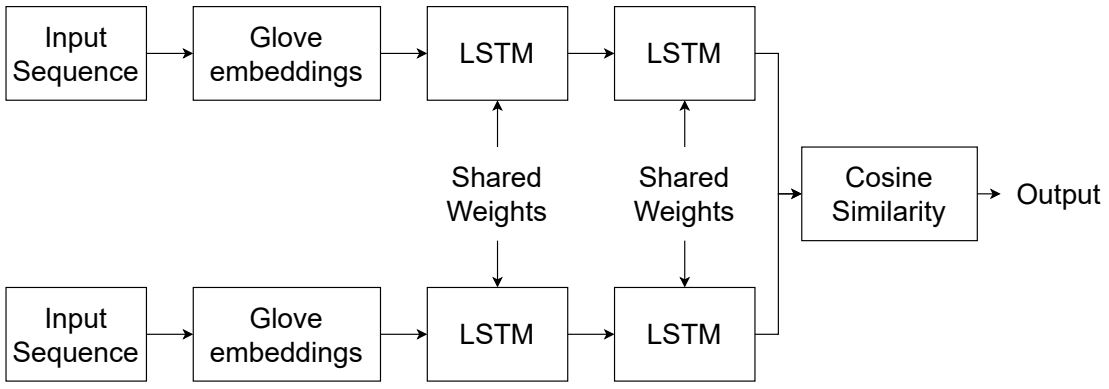
Model BERT je navrhnutý tak, že veta musí začínať tokenom [CLS] a končiť tokenom [SEP]. Kde keď pracujeme s viacerými vetami na vstupe tak je potreba oddeľovať vety tokenom [SEP], ale vďaka knižnici Hugging-face to knižnica tokenizerov robí za nás. Vďaka tejto vlastnosti sme okrem Siamese BERT trénovali aj samotný BERT ktorých výsledky možno vidieť v sekcii 5



Obrázek 1: Schéma modelu Siamese BERT

4.2 LSTM

Rovnako ako v predchádzajúcej sekcií každá vstupná sekvencia predstavuje jednú otázku. Tie sú posielané na vstup Glove embeddingu ktorý nám dá vektory reprezentujúce slová vo vete, tie sa ďalej posielajú cez dve obojsmerné skryté vrstvy LSTM sieti a výsledky sa porovnávajú pomocou Cosine similarity. Každá s LSTM vrstviev ma hodnotu `hidden_size` nastavenú na 50.

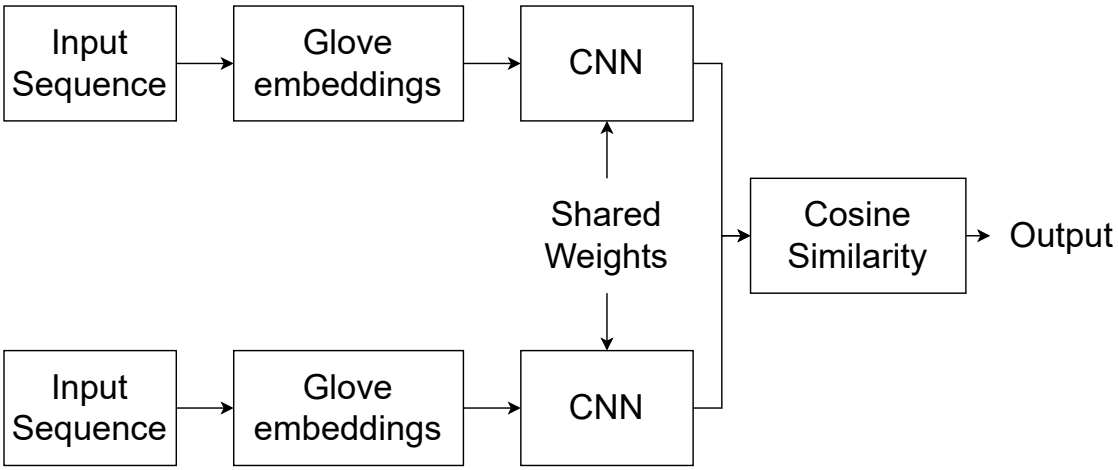


Obrázek 2: Schéma modelu Siamese LSTM

4.3 CNN

Rovnako ako v 4.2, boli vstupné sekvencie posieláne do Glove embeddings pre reprezentáciu slov vektormi. Glove embeddings boli v každom modeli trénované zároveň s modelom. Následne boli vektory posielané konvolučných

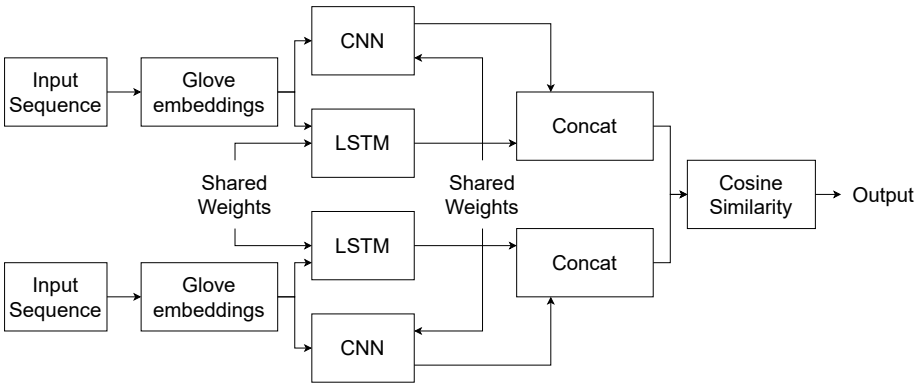
neurónových sietí, skladajúcich sa z 4 vrstiev konvolúcie nasledovaných aktivačnými vrstvami, kde za každou vrstvou konvolúcie nasleduje aktivačná funkcia, pričom veľkosti filtrov jednotlivých konvolučných vrstiev sú, dvakrát 5×100 a dvakrát 3×100. Pre zníženie tzv. 'overfittingu' sme nakoniec pridali jednu `dropout` vrstvu s šancou `0.3`.



Obrázek 3: Schéma modelu Siamese CNN

4.4 LSTM CNN

Ako môžeme pozorovať na obrázku 4, LSTMCNN skladá sa z dvoch CNN so zdielanými váhami a dvoch LSTM taktiež s zdielanými váhami, kde vstupy sú rovnaké ako v predchádzajúcich modeloch, a výstupy týchto LSTM a CNN sú spájané a posielané do vrstvy pre porovnanie podobnosti. V tomto modele sa konvolučné siete skladajú z jednej vrstvy konvolúcie, aktivačnej ReLU funkcie nasledovanej 'dropout' vrstvou pre zníženie šance na 'overfitting', pričom parametre konvolučného filtra sú 5×100 a šanca pre 'dropout' je nastavená na `0.3`.



Obrázek 4: Schéma modelu Siamese LSTMCNN

5 Vyhodnotenie

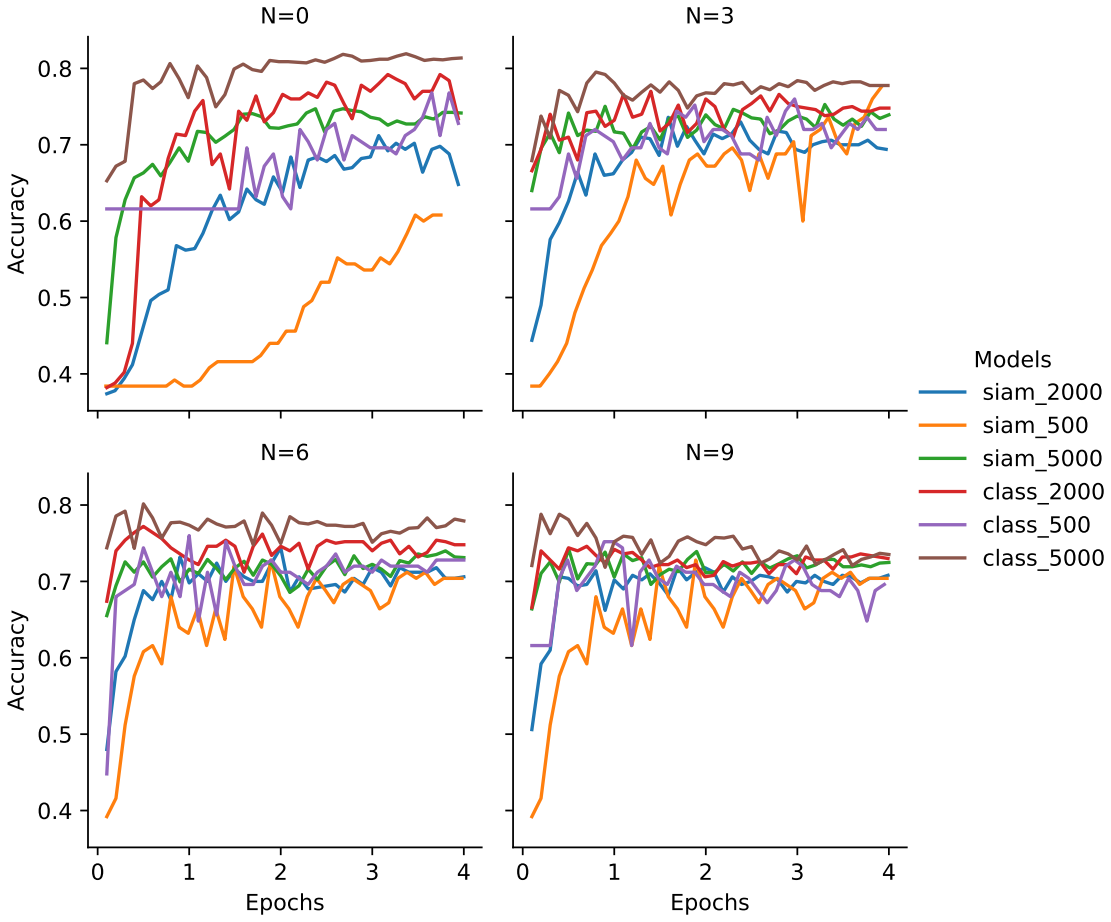
Boli vykonané dve sady testov, prvá sada bola zameraná na skúmanie vplyvu augmentácie na celkovú presnosť siete pri použití iba čiastkovej trénovacej sady a druhá sada skúmala celkovú presnosť sietí po natrénovaní na celkom datasete.

5.1 Vyhodnotenie augmentácie

5.1.1 BERT

Oba modely typu BERT (`Siamese` a `Classifier`) boli dotrénované po dobu 4 epoch na štyroch rôznych veľkostiach trénovacej sady `size_of_train = [500, 2000, 5000]` rovnako ako v štúdií o augmentácií. Takisto

boli pre každú trénovaciu sadu vyskúšané 4 rôzne intenzity augmentácie `aug_intensity`, a to `0` - žiadna augmentácia, slúžiaca ako základ pre porovnávanie a následne `3`, `6`, `9` (číslo reprezentuje počet nových otázkových párov vzniknutých po augmentácií). Ako strátová funkcia bola použitá `MeanSquareError`, ako optimiser `Adam` a na overenie celkovej presnosti sme použili metriku `accuracy_score` z knižnice `sklearn`.

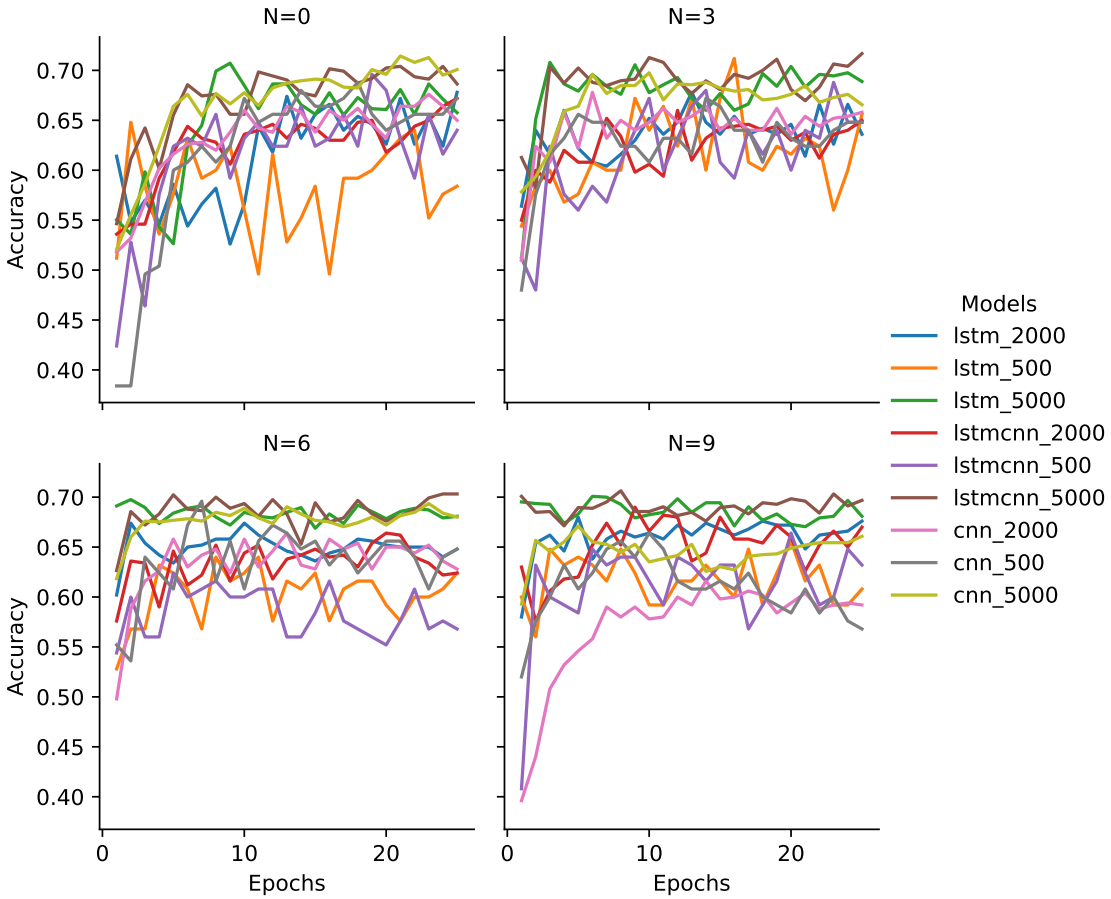


Obrázek 5: Hodnota `accuracy_score` vypočítaná na evaluačnej sade počas trénovania modelov BERT v jednotlivých epochách

Z grafov znázornených na obrázku 5 je možné si všimnúť, že akákoľvek augmentácia mala pri väčšom množstve trénovacích vzoriek v najlepšom prípade neutrálny efekt a v tom horšom viedla k zníženiu celkovej presnosti siete typu BERT. Pri malom množstve dát je možné si povšimnúť rýchlejšiu konvergenciu a miestami vyššiu presnosť sietí trénovaných na augmentovaných dátach, pričom najlepšie výsledky dosahovali siete, pri ktorých bola intensita augmentácie `aug_intensity` nastavená na hodnotu `3`.

5.1.2 Ostatné modely

Pred tým, než bude možné prehlásiť, že spôsob augmentácie implementovaný v tomto projekte nemá pozitívny vplyv na presnosť predikcie podobnosti otázok, bolo nutné ho ešte vyskušať na modeloch, ktoré neboli predom natrénované na vysokom množstve dát. Preto sme rovnaké testy vykonali aj na menších modeloch `LSTM`, `CNN` a `LSTMCNN`, ktoré boli natrénované po dobu 25 epoch. Zvyšné parametre trénovania sú identické s trénovaním modelov typu BERT.



Obrázek 6: Hodnota `accuracy_score` vypočítaná na evaluačnej sade počas trénovania siamských modelov v jednotlivých epochách

Z výsledkov grafov na obrázku 6 vyplýva, že narozdiel od modelov BERT použitie augmentácie pri menších modeloch prinieslo zrýchlenie konvergencie a malé zlepšenie presnosti na evaluačnej sade aj s väčším množstvom trénovacích dát. Zároveň sa potvrdilo predošlé pozorovanie, že nižšia úroveň intenzity augmentácie (`aug_intensity` = 3) dosahuje lepších výsledkov, keďže pri vyšších hodnotách už nastáva 'overfitting' na trénovacích dátach.

5.1.3 Celkové výsledky

V konečnom dôsledku však na základe dát z tabuľky 7 môžeme vyvodiť, že augmentácia mala neutrálny až miestami negatívny vplyv na celkovú presnosť siete na testovacích dátach - či už pri dotrénovaných modeloch typu BERT, alebo nami implementovaných modeloch `LSTM`, `LSTMCNN`, `CNN`.

Samples	Aug. intensity	siam_bert	class_bert	siam_cnn	siam_lstm	siam_lstmcnn
500	0	0.59	0.73	0.59	0.43	0.64
	3	0.66	0.72	0.64	0.59	0.64
	6	0.67	0.71	0.61	0.62	0.64
	9	0.68	0.70	0.60	0.64	0.63
2000	0	0.69	0.77	0.69	0.67	0.67
	3	0.69	0.75	0.66	0.64	0.68
	6	0.70	0.75	0.63	0.65	0.66
	9	0.70	0.74	0.59	0.65	0.66
5000	0	0.74	0.81	0.70	0.65	0.69
	3	0.72	0.76	0.65	0.67	0.68
	6	0.72	0.76	0.65	0.67	0.69
	9	0.72	0.76	0.65	0.67	0.67

Tabulka 7: Ukážka hodnoty `accuracy_score` na testovacej sade pri rôznych veľkostiach trénovacej sady a rôznych intenzitách augmentácie

5.2 Výsledky našich modelov

Aby sme mohli porovnať účinnosť modelov trénovaných na malej vzorke dát, bolo nutné natrénovať nami navrhnuté modely aj na plnej trénovacej sade (cca. 300000 vzoriek). Modely boli trénované po dobu 25 epoch, pôvodná trénovacia sada bola rozdelená v pomere 75:25 na trénovaciu a evaluačnú sadu. Strátovú funkciu, metriku presnosti a optimiser sme použili rovnaký ako v predošlom testovaní.

Model	F1 score	Accuracy
LSTM	0.73	0.80
CNN	0.76	0.80
LSTMCNN	0.75	0.80

Tabulka 8: Ukážka hodnoty `accuracy_score` a `F1` skóre na testovacej sade pri maximálnej veľkosti trénovacej sady

6 Záver

Výsledkom projektu je komplexný nástroj na trénovanie modelov rôznych architektúr, pre klasifikovanie podobnosti viet pre Quora question pairs dataset, obsahujúci preprocessing, trénovacie skripty, augmentáciu a validačný skript. Pre dosiahnutie čo najlepších výsledkov boli testované modely s rôznymi modifikáciami a skupina augmentačných metód. Ako celkový víťaz skončil pôvodný BERT bez modifikácií. Augmentácia dát sa ukázala ako neafektívna voľba, jedine v prípade nedostatočného množstva dát mal daný spôsob augmentácie pozitívny vplyv, a aj to z dôvodu šíreho nedostatku dát pre natrénovanie daného modelu.

Reference

[Fac22] Hugging Face. transformers, 2022. [Online; accessed 28-5-2022].

[Kag07] Kaggle. Quora Question Pairs, 2007. [Online; accessed 27-5-2022].

[NLT22] NLTK. Sample usage for wordnet, 2022. [Online; accessed 27-5-2022].

[Wik21] Wikipedia. Stop word, 2021. [Online; accessed 27-5-2022].

[WZ19] Jason Wei and Kai Zou. Eda: Easy data augmentation techniques for boosting performance on text classification tasks, 2019.