

[NLP]

Dokumentacja Końcowa

Piotr Frątczak
(300207)

Kacper Maj
(300210)

1 czerwca 2022

1 Zadanie

Rekurencyjna głęboka sieć neuronowa (konkatenacja wektorów).

Konkatenacja wektora reprezentującego POS z wektorem word embedding reprezentującym słowo. Wynikający skonkatenowany wektor jest wejściem dla sieci. Należy przeprowadzić eksperymenty dla kilku zbiorów. Porównać wyniki dla podstawowej architektury RNN i architektury ze znacznikiem POS.

2 Architektura

W celu implementacji sieci zostało wykorzystane rozwiązanie przygotowane przez inny zespół. W związku z tym zbadano kilka wariantów modelu. Będą one mogły różniły się przede wszystkim sposobem wykorzystania i typem znaczników POS - Universal lub Penn Treebank. W testach uwzględniono także dodatkowy typ sieci, bazujący na modelu RoBERTa.

Wykorzystane dotychczasowe modele:

- LSTM_Single_Cell - model ten składa się tylko z pojedynczej komórki LSTM,
- LSTM_Layer - w skład tego modelu wchodzi wiele standardowych komórek LSTM,
- LSTM_POS_Universal - składa się z komórek LSTM, przy czym każda odpowiada pojedynczemu znacznikowi Universal,
- LSTM_POS_Penn - model ten wykorzystuje znaczniki POS Penn Treebank, każdemu ze znaczników odpowiada oddzielna komórka LSTM,
- Roberta Classifier - oparty o model Roberta base, jednak z dwoma dodatkowymi w pełni połączonymi warstwami, pozwalającymi na dostosowanie modelu Roberta do badanych zbiorów danych. Wykorzystana została implementacja dostępna w repozytorium torch.hub.

Dodane modele:

- LSTM_Concat_Penn - architektura, gdzie graf dynamiczny zastąpiono konkatenacją wektorów zanurzonych tokenów i wektorów reprezentujących POS, także zanurzonych. W tym wypadku użyto znaczników Penn Treebank.
- LSTM_Concat_Universal - jak wyżej, lecz wykorzystano znaczniki POS Universal.
- LSTM_Concat_Dependency - poza znacznikiem POS, do zanurzenia dokenu doklejony został również typ rozkładu zależnościowego. Znacznik oznaczający typ z rozkładu zależnościowego doklejano do

lewego słowa w relacji, tzn. po rozkładzie zależnościowym otrzymywano drzewo i doklejało znacznik do rodzica.

3 Eksperymenty

Eksperymenty zostały przeprowadzone na pięciu zbiorach danych po czym wyniki zostały posortowane po wartości F1 i na tej podstawie ocenione. Zdecydowaliśmy się na używanie metryki F1 z powodu obawy o niezbalansowanie klas. Niestety dostarczony projekt wybierał najlepszy model na podstawie `test_loss`. Modyfikacja tej cechy wykraczała poza założenia projektu, w związku z tym nie zdecydowaliśmy się na nią.

3.1 Zbiory Danych

Eksperymenty przeprowadzono na poniższych zbiorach danych:

- **BBC Datasets** - Zbiór artykułów z BBC oznaczonych kategoriami (np. sport, newsy itd.)
- **SMS Spam Collection Dataset** - Zbiór wiadomości SMS oznaczonych jako spam lub nie.
- **NLP with Disaster Tweets** - Zbiór Tweetów o prawdziwych katastrofach lub nie.
- **Emotions dataset for NLP** - Zbiór dokumentów tekstowych z przypisanymi emocjami.
- **IMDB Dataset of 50K Movie Reviews** - Zbiór recenzji filmów z IMDB z przypisanym oznaczeniem czy opinia jest pozytywna czy negatywna.

3.1.1 Przegląd Literatury

W zadaniach typu machine reading znane jest podejście reprezentacji słów jako konkatenacja wektorów zanurzenia (embedding, np. GloVe) oraz POS tagu. Dodatkowo dodaje się także wektorowe reprezentacje takich elementów jak NER, exact match i align question embedding [2, 3].

W [1] przedstawiono standardowe zanurzenia słów oraz udoskonalenia (konkatenacja słów i znaczników POS), które wprowadzone w generowaniu zanurzonych słów, aby uchwycić morfologię na podstawie ciągów ortograficznych (w zadaniu continuous-state parsing).

W artykule rozwiązujący problem czy opinia dotycząca linii lotniczych w pojedynczym Tweecie jest pozytywna czy negatywna (zadanie podobne do NLP with Disaster Tweets) dokonano konkatenacji słów i znaczników POS oraz innych elementów w celu stworzenia głębokiego, inteligentnego zanurzenia kontekstowego. Połączono osadzanie kontekstowe (ELMo), osadzanie słów (GloVe), osadzanie części mowy (POS) i osadzanie leksykonów. Brak POS spowodował duży spadek miary accuracy w ablation test. Wykorzystany model również był oparty na LSTM, jednak bardziej rozbudowany - BiLSTM z atencją [4].

3.2 Wyniki

Tablice podzielono ze względu na zbiory danych.

Legenda do tablic:

sl	sequence_length
tl	test_loss
es	embedding_size
lr	learning_rate
pad	padding

3.2.1 Oryginalne architektury

accuracy	precision	recall	f1	tl	model	sl	es	epochs	lr	pad
0.946	0.947	0.946	0.946	0.957	LSTM_Single_Cell	200	50	30	0.001	pre
0.935	0.936	0.935	0.935	0.965	LSTM_Layer	200	100	25	0.001	pre
0.933	0.933	0.933	0.933	0.969	LSTM_Single_Cell	50	100	30	0.001	pre
0.926	0.928	0.926	0.925	0.969	LSTM_Single_Cell	200	100	24	0.001	pre
0.886	0.893	0.886	0.887	1.017	LSTM_POS_Universal	200	100	29	0.001	pre
0.888	0.894	0.888	0.887	0.987	LSTM_Single_Cell	50	50	30	0.001	pre
0.857	0.863	0.857	0.856	1.020	LSTM_POS_Penn	50	100	24	0.001	pre
0.818	0.830	0.818	0.817	1.063	LSTM_POS_Universal	50	50	30	0.001	pre
0.724	0.726	0.724	0.725	1.283	Roberta Classifier	200		28	0.001	
0.771	0.843	0.771	0.720	1.033	LSTM_POS_Universal	50	100	29	0.001	pre
0.742	0.759	0.742	0.689	1.058	LSTM_Layer	50	100	25	0.001	pre
0.655	0.704	0.655	0.649	1.237	LSTM_POS_Penn	200	100	18	0.001	pre
0.599	0.607	0.599	0.600	1.350	Roberta Classifier	50		29	0.001	
0.657	0.572	0.657	0.588	1.195	LSTM_POS_Universal	200	50	29	0.001	pre
0.617	0.624	0.617	0.563	1.164	LSTM_POS_Penn	200	50	29	0.001	pre
0.641	0.720	0.641	0.563	1.091	LSTM_Layer	200	50	30	0.001	pre
0.567	0.399	0.567	0.457	1.175	LSTM_Layer	50	50	27	0.001	pre
0.538	0.353	0.538	0.420	1.216	LSTM_POS_Penn	50	50	8	0.001	pre

Tablica 1: BBC_news

accuracy	precision	recall	f1	tl	model	sl	es	epochs	lr	pad
0.973	0.973	0.973	0.973	0.338	LSTM_POS_Universal	50	100	23	0.001	pre
0.973	0.973	0.973	0.972	0.338	LSTM_Single_Cell	50	100	23	0.001	pre
0.969	0.969	0.969	0.969	0.345	LSTM_POS_Universal	50	50	26	0.001	pre
0.968	0.967	0.968	0.967	0.346	LSTM_POS_Penn	200	100	23	0.001	pre
0.966	0.965	0.966	0.965	0.344	LSTM_POS_Penn	50	100	30	0.001	pre
0.963	0.962	0.963	0.963	0.348	LSTM_POS_Universal	200	100	30	0.001	pre
0.961	0.959	0.961	0.959	0.354	LSTM_POS_Penn	50	50	29	0.001	pre
0.951	0.949	0.951	0.950	0.359	LSTM_POS_Universal	200	50	26	0.001	pre
0.876	0.913	0.876	0.888	0.627	Roberta Classifier	50		17	0.001	
0.872	0.927	0.872	0.887	0.622	Roberta Classifier	200		8	0.001	
0.869	0.755	0.869	0.808	0.444	LSTM_Layer	50	50	24	0.001	pre
0.869	0.755	0.869	0.808	0.444	LSTM_Single_Cell	50	50	24	0.001	pre
0.869	0.755	0.869	0.808	0.445	LSTM_Layer	50	100	27	0.001	pre
0.869	0.755	0.869	0.808	0.445	LSTM_Layer	200	50	18	0.001	pre
0.869	0.755	0.869	0.808	0.444	LSTM_Single_Cell	200	50	26	0.001	pre
0.869	0.755	0.869	0.808	0.445	LSTM_POS_Penn	200	50	26	0.001	pre
0.869	0.755	0.869	0.808	0.444	LSTM_Layer	200	100	24	0.001	pre
0.869	0.755	0.869	0.808	0.444	LSTM_Single_Cell	200	100	26	0.001	pre

Tablica 2: SpamSMS

accuracy	precision	recall	f1	tl	model	sl	es	epochs	lr	pad
0.756	0.746	0.756	0.747	0.549	LSTM_Layer	50	50	12	0.001	pre
0.750	0.740	0.750	0.741	0.549	LSTM_Layer	200	50	13	0.001	pre
0.750	0.739	0.750	0.739	0.552	LSTM_Single_Cell	200	50	18	0.001	pre
0.747	0.736	0.747	0.738	0.554	LSTM_Layer	50	100	9	0.001	pre
0.714	0.750	0.725	0.737	0.642	Roberta Classifier	200		14	0.001	
0.748	0.736	0.748	0.734	0.556	LSTM_Layer	200	100	10	0.001	pre
0.745	0.733	0.745	0.733	0.554	LSTM_Single_Cell	200	100	12	0.001	pre
0.744	0.731	0.744	0.731	0.559	LSTM_Single_Cell	50	50	13	0.001	pre
0.746	0.733	0.746	0.729	0.558	LSTM_POS_Universal	50	50	30	0.001	pre
0.743	0.730	0.743	0.729	0.558	LSTM_Single_Cell	50	100	12	0.001	pre
0.743	0.729	0.743	0.726	0.562	LSTM_POS_Universal	50	100	25	0.001	pre
0.734	0.721	0.734	0.723	0.561	LSTM_POS_Universal	200	50	24	0.001	pre
0.739	0.725	0.739	0.722	0.565	LSTM_POS_Penn	200	50	30	0.001	pre
0.741	0.727	0.741	0.721	0.563	LSTM_POS_Penn	50	100	28	0.001	pre
0.735	0.720	0.735	0.717	0.567	LSTM_POS_Universal	200	100	28	0.001	pre
0.733	0.718	0.733	0.714	0.564	LSTM_POS_Penn	200	100	19	0.001	pre
0.710	0.713	0.706	0.710	0.651	Roberta Classifier	50		10	0.001	
0.735	0.721	0.735	0.707	0.568	LSTM_POS_Penn	50	50	30	0.001	pre

Tablica 3: DisasterTweets

accuracy	precision	recall	f1	tl	model	sl	es	epochs	lr	pad
0.922	0.923	0.923	0.922	1.100	LSTM_Layer	200	100	30	0.001	pre
0.902	0.870	0.903	0.886	1.122	LSTM_Layer	50	100	28	0.001	pre
0.885	0.888	0.885	0.882	1.133	LSTM_Layer	200	50	30	0.001	pre
0.898	0.867	0.898	0.881	1.126	LSTM_Single_Cell	50	100	30	0.001	pre
0.898	0.866	0.898	0.881	1.125	LSTM_Single_Cell	200	100	25	0.001	pre
0.879	0.883	0.880	0.877	1.145	LSTM_Single_Cell	200	50	29	0.001	pre
0.865	0.869	0.865	0.850	1.161	LSTM_POS_Universal	200	100	29	0.001	pre
0.866	0.835	0.867	0.850	1.154	LSTM_Layer	50	50	30	0.001	pre
0.840	0.834	0.836	0.835	1.373	Roberta Classifier	200		8	0.001	
0.842	0.815	0.842	0.826	1.180	LSTM_Single_Cell	50	50	28	0.001	pre
0.835	0.806	0.835	0.819	1.190	LSTM_POS_Universal	50	100	29	0.001	pre
0.802	0.774	0.802	0.787	1.216	LSTM_POS_Penn	50	100	29	0.001	pre
0.792	0.772	0.803	0.787	1.420	Roberta Classifier	50		14	0.001	
0.798	0.769	0.799	0.783	1.220	LSTM_POS_Penn	200	100	29	0.001	pre
0.738	0.747	0.748	0.725	1.266	LSTM_POS_Universal	50	50	30	0.001	pre
0.678	0.656	0.683	0.652	1.328	LSTM_POS_Penn	200	50	30	0.001	pre
0.674	0.659	0.675	0.640	1.319	LSTM_POS_Universal	200	50	29	0.001	pre
0.669	0.642	0.669	0.632	1.329	LSTM_POS_Penn	50	50	29	0.001	pre

Tablica 4: Emotions

accuracy	precision	recall	f1	tl	model	sl	es	epochs	lr	pad
0.762	0.762	0.762	0.762	0.546	LSTM_Single_Cell	50	100	13	0.001	pre
0.758	0.761	0.758	0.757	0.543	LSTM_POS_Universal	50	100	24	0.001	pre
0.754	0.754	0.754	0.754	0.554	LSTM_Layer	50	100	8	0.001	pre
0.749	0.749	0.749	0.749	0.547	LSTM_POS_Universal	50	50	29	0.001	pre
0.743	0.746	0.746	0.746	0.683	Roberta Classifier	50		11	0.001	
0.743	0.744	0.743	0.743	0.564	LSTM_Layer	200	100	10	0.001	pre
0.730	0.740	0.742	0.741	0.667	Roberta Classifier	200		13	0.001	
0.736	0.736	0.736	0.736	0.558	LSTM_POS_Penn	50	50	30	0.001	pre
0.736	0.736	0.736	0.736	0.569	LSTM_Single_Cell	50	50	9	0.001	pre
0.733	0.733	0.733	0.733	0.559	LSTM_POS_Penn	50	100	28	0.001	pre

Tablica 5: IMDB_reviews

Należy tutaj wyszczególnić dwa rozwiązania: LSTM_Single_Cell i LSTM_Layer. LSTM_Single_Cell ($sl = 50$, $cs = 100$) w prawie każdym zbiorze danych osiąga wyniki lepsze niż znaczna większość sprawdzanych modeli. LSTM_Layer ($sl = 200$, $cs = 100$) osiąga nieznacznie odbiegające wyniki lecz w zbiorze BBC_news lepsze okazuje się to samo rozwiązanie z wartościami $sl = 200$ i $cs = 100$ ponad to ustępuje innym modelom w zbiorze DisasterTweets. Jednak zasadność brania pod uwagę tego zbioru jako wyznacznika jest wątpliwa. Wskaźnik F1 w tym zbiorze spada z najlepszego do najgorszego modelu o tylko 0.040 punktu. Podobnie sprawa wygląda ze zbiorem danych IMDB_reviews gdzie różnica wynosi tylko 0.029 punktu.

Najgorszym rozwiązaniem wydaje się LSTM_POS_Penn. Nie licząc SpamSMS wypada ono najgorzej zarówno w przypadku najlepiej dobranych dla modelu wartości sl i cs oraz uśredniając wszystkie rozwiązania bez względu na sl i cs . W zbiorze SpamSMS lepiej radzą sobie, gorzej wyglądają wszędzie indziej LSTM_POS_Penn i LSTM_POS_Universal. Nie licząc LSTM_POS_Penn ($sl = 200$, $cs = 50$) osiągają one $F1 \geq 0.95$ podczas gdy inne rozwiązania nie przekraczają 0.9. Niezależnie od rozwiązania najlepsze wydają się wartości $sl = 50$ $cs = 100$. Zazwyczaj generują one najlepszy lub zbliżony do najlepszego model w każdym z rozwiązań. Jedynym odstępstwem godnym uwagi jest zbiór danych Emotions gdzie LSTM_Layer o wartościach $sl = 200$ $cs = 100$ wygenerował model o $F1 = 0.922$, a LSTM_Layer o wartościach $sl = 50$ $cs = 100$ wygenerował model o $F1 = 0.886$.

Na koniec warto też wspomnieć o rozwiązaniu Roberta Classifier. Jest to zdecydowanie najdłużej uczące się rozwiązanie (długość uczenia jest kilkunastokrotnie większa od innych rozwiązań). Najlepsze wyniki osiąga on po ok. dwukrotnie mniejszej ilości iteracji niż reszta modeli, jednak nawet skrócenie czasu o połowę klasyfikuje go jako najdłużej uczące się rozwiązanie. Długość uczenia może być spowodowana złą implementacją modelu. Pomimo długiego uczenia wydaje się on klasyfikować zawsze w środkowej części tabeli. Jest to też jedyne rozwiązanie, które mogłoby zyskać na zmianie oceniania z test_loss na F1 podczas wybierania najlepszego modelu.

3.2.2 Architektura z konkatenacją POS

accuracy	precision	recall	f1	tl	model	sl	es	epochs	lr	pad
0.868	0.886	0.868	0.864	1.026	LSTM_Concat_Penn	50	100	28	0.001	pre
0.805	0.847	0.805	0.807	1.079	LSTM_Concat_Penn	200	100	22	0.001	pre
0.617	0.618	0.617	0.509	1.118	LSTM_Concat_Universal	50	100	30	0.001	pre
0.209	0.043	0.209	0.072	1.570	LSTM_Concat_Universal	200	100	29	0.001	pre
0.209	0.043	0.209	0.072	1.605	LSTM_Concat_Dependency	50	100	6	0.001	pre
0.209	0.043	0.209	0.072	1.604	LSTM_Concat_Dependency	200	100	8	0.001	pre

Tablica 6: BBC_news

accuracy	precision	recall	f1	tl	model	sl	es	epochs	lr	pad
0.869	0.755	0.869	0.808	0.445	LSTM_Concat_Penn	50	100	25	0.001	pre
0.869	0.755	0.869	0.808	0.444	LSTM_Concat_Penn	200	100	30	0.001	pre
0.869	0.755	0.869	0.808	0.444	LSTM_Concat_Universal	200	100	29	0.001	pre
0.869	0.755	0.869	0.808	0.445	LSTM_Concat_Universal	50	100	29	0.001	pre
0.869	0.755	0.869	0.808	0.445	LSTM_Concat_Dependency	50	100	8	0.001	pre
0.869	0.755	0.869	0.808	0.445	LSTM_Concat_Dependency	200	100	2	0.001	pre

Tablica 7: SpamSMS

accuracy	precision	recall	f1	tl	model	sl	es	epochs	lr	pad
0.749	0.738	0.749	0.739	0.559	LSTM_Concat_Penn	50	100	14	0.001	pre
0.754	0.743	0.754	0.735	0.558	LSTM_Concat_Universal	200	100	11	0.001	pre
0.741	0.731	0.741	0.733	0.569	LSTM_Concat_Penn	200	100	19	0.001	pre
0.745	0.733	0.745	0.731	0.561	LSTM_Concat_Universal	50	100	10	0.001	pre
0.736	0.724	0.736	0.706	0.565	LSTM_Concat_Dependency	200	100	4	0.001	pre
0.682	0.466	0.682	0.554	0.627	LSTM_Concat_Dependency	50	100	7	0.001	pre

Tablica 8: DisasterTweets

accuracy	precision	recall	f1	tl	model	sl	es	epochs	lr	pad
0.910	0.908	0.910	0.907	1.107	LSTM_Concat_Universal	200	100	30	0.001	pre
0.900	0.870	0.901	0.884	1.123	LSTM_Concat_Universal	50	100	27	0.001	pre
0.893	0.865	0.893	0.876	1.125	LSTM_Concat_Penn	50	100	29	0.001	pre
0.888	0.857	0.889	0.873	1.132	LSTM_Concat_Penn	200	100	29	0.001	pre
0.343	0.296	0.343	0.236	1.636	LSTM_Concat_Dependency	50	100	9	0.001	pre
0.291	0.085	0.291	0.131	1.587	LSTM_Concat_Dependency	200	100	5	0.001	pre

Tablica 9: Emotions

accuracy	precision	recall	f1	tl	model	sl	es	epochs	lr	pad
0.852	0.852	0.852	0.852	0.459	LSTM_Concat_Penn	200	100	17	0.001	pre
0.850	0.850	0.850	0.849	0.462	LSTM_Concat_Universal	200	100	9	0.001	pre
0.759	0.760	0.759	0.758	0.548	LSTM_Concat_Universal	50	100	7	0.001	pre
0.754	0.755	0.754	0.754	0.554	LSTM_Concat_Penn	50	100	6	0.001	pre
0.732	0.738	0.732	0.730	0.563	LSTM_Concat_Dependency	50	100	6	0.001	pre
0.718	0.724	0.718	0.716	0.577	LSTM_Concat_Dependency	200	100	7	0.001	pre

Tablica 10: IMDB_reviews

Należy zwrócić uwagę, że LSTM_Concat (nie licząc LSTM_Concat_Dependency $sl = 50$ $cs = 100$) w zbiorze danych DisasterTweets klasyfikuje się pomiędzy najlepszym, a najgorszym modelem potwierdzając, że ten zbiór nie jest dobrym wyznacznikiem jakości, jeśli tylko zbiór osiąga w nim zadowalające wyniki.

Zauważalną poprawę widać natomiast w zbiorze IMDB_reviews. LSTM_Concat_Penn $sl = 200$ $cs = 100$ i LSTM_Concat_Universal $sl = 200$ $cs = 100$ zwiększyły wartość F1 kolejno o 0.090 i 0.087 względem poprzedniego najlepszego modelu. Zbiorem, w którym wyniki pogorszyły się jest zbiór SpamSMS. Każdy model w tym zbiorze osiągnął najgorsze możliwe wyniki. Dokładnie te same wyniki zostały osiągnięte w poprzedniej architekturze przez połowę modeli. Jest to spowodowane prawdopodobnie tym, że w zbiorze występuje klasyfikacja binarna i najgorsze modele klasyfikują wszystkie dane jako 1 lub 0 niezależnie od atrybutów.

Nie zauważono żadnej zależności pomiędzy wartościami sl i cs , a jakością modelu jak w poprzednich modelach. Dobór tych wartości powinien być dostosowany w zależności od zbioru danych.

Należy też wspomnieć o najgorzej wyglądającym rozwiązaniu LSTM_Concat_Dependency. Jest to rozwiązanie mające najbardziej złożoną reprezentację słów (w przeciwieństwie do LSTM_Concat_Penn i LSTM_Concat_Universal oprócz POS do słowa dodawany jest także typ rozbioru zależnościowego). Wydawałoby się, że najbardziej bogate w informację rozwiązanie powinno być najlepsze, lecz eksperymenty tego nie potwierdzają. Istnieje też szansa, że implementacja lub wykorzystanie typu rozbioru zależnościowego są niewłaściwe. Szybkość uczenia tego modelu jest podobna do rozwiązania Roberta Classifier, a najlepszy model jest osiągany również po mniejszej ilości iteracji niż w pozostałych rozwiązaniach.

3.3 Konkluzja

W porównaniu z poprzednimi modelami architektura ze znacznikiem POS nie ma przewagi nad innymi rozwiązaniami. Lepiej radzi sobie w zbiorze IMDB_reviews, lecz jest to zrównoważone przez bycie najgorszymi rozwiązaniami w zbiorze SpamSMS. W pozostałych przypadkach najlepsze rozwiązania klasyfikują się w środku tabel lub jeśli będziemy wybierać z nich najgorsze modele znacznie pogarszają wyniki (LSTM_Concat_Dependency pogorszył najgorszy wynik F1 o 0.348 w zbiorze BBC_news).

Na koniec porównajmy po dwa najlepsze modele z starej i nowej architektury

accuracy	precision	recall	f1	tl	model	sl	es	epochs	lr	pad
0.935	0.936	0.935	0.935	0.965	LSTM_Layer	200	100	25	0.001	pre
0.933	0.933	0.933	0.933	0.969	LSTM_Single_Cell	50	100	30	0.001	pre
0.868	0.886	0.868	0.864	1.026	LSTM_Concat_Penn	50	100	28	0.001	pre
0.209	0.043	0.209	0.072	1.570	LSTM_Concat_Universal	200	100	29	0.001	pre

Tablica 11: BBC_news

accuracy	precision	recall	f1	tl	model	sl	es	epochs	lr	pad
0.973	0.973	0.973	0.972	0.338	LSTM_Single_Cell	50	100	23	0.001	pre
0.869	0.755	0.869	0.808	0.444	LSTM_Layer	200	100	24	0.001	pre
0.869	0.755	0.869	0.808	0.445	LSTM_Concat_Penn	50	100	25	0.001	pre
0.869	0.755	0.869	0.808	0.444	LSTM_Concat_Universal	200	100	29	0.001	pre

Tablica 12: SpamSMS

accuracy	precision	recall	f1	tl	model	sl	es	epochs	lr	pad
0.922	0.923	0.923	0.922	1.100	LSTM_Layer	200	100	30	0.001	pre
0.910	0.908	0.910	0.907	1.107	LSTM_Concat_Universal	200	100	30	0.001	pre
0.893	0.865	0.893	0.876	1.125	LSTM_Concat_Penn	50	100	29	0.001	pre
0.898	0.867	0.898	0.881	1.126	LSTM_Single_Cell	50	100	30	0.001	pre

Tablica 13: Emotions

accuracy	precision	recall	f1	tl	model	sl	es	epochs	lr	pad
0.850	0.850	0.850	0.849	0.462	LSTM_Concat_Universal	200	100	9	0.001	pre
0.762	0.762	0.762	0.762	0.546	LSTM_Single_Cell	50	100	13	0.001	pre
0.754	0.755	0.754	0.754	0.554	LSTM_Concat_Penn	50	100	6	0.001	pre
0.743	0.744	0.743	0.743	0.564	LSTM_Layer	200	10	10	0.001	pre

Tablica 14: IMDB_reviews

Porównanie, które proponujemy w ocenie tych modeli to punktacja za osiągnięte miejsca:

1. miejsce - 4 punkty
2. miejsce - 3 punkty
3. miejsce - 2 punkty
4. miejsce - 1 punkt

Po policzeniu punktów zajęte miejsca wyglądają następująco:

1. LSTM_Layer $sl = 200$, $cs = 100$ - 12 punktów
2. LSTM_Single_Cell $sl = 50$, $cs = 100$ - 11 punktów
3. LSTM_Concat_Universal $sl = 200$, $cs = 100$ - 9 punktów
4. LSTM_Concat_Penn $sl = 50$, $cs = 100$ - 8 punktów

To tylko potwierdza całkowity brak przewagi nad starą architekturą.

4 Implementacja


4.1 Środowisko

Język programowania: Python 3.8.

Biblioteki:

- Keras - narzędzia do tokenizacji korpusu.
- NLTK - zawiera znaczniki *Penn Treebank* i *Universum*.
- Magnitude - wektorowa reprezentacja znaczników POS i typów rozkładu zależnościowego.
- Stanza - rozkład zależnościowy tekstów.
- PyTorch - Implementacja LSTM oraz reprezentacja słów w postaci wektorowej.

4.2 Repozytorium

Kod źródłowy jest dostępny na:  `ornn-pos`

Bibliografia

- [1] Miguel Ballesteros, Chris Dyer i Noah A Smith. „Improved transition-based parsing by modeling characters instead of words with LSTMs”. W: *arXiv preprint arXiv:1508.00657* (2015).
- [2] Danqi Chen i in. „Reading wikipedia to answer open-domain questions”. W: *arXiv preprint arXiv:1704.00051* (2017).
- [3] Xiaodong Liu i in. „Stochastic answer networks for machine reading comprehension”. W: *arXiv preprint arXiv:1712.03556* (2017).
- [4] Usman Naseem i Katarzyna Musiał. „Dice: Deep intelligent contextual embedding for twitter sentiment analysis”. W: *2019 International conference on document analysis and recognition (ICDAR)*. IEEE. 2019, s. 953–958.