Politechnika Warszawska

WYDZIAŁ ELEKTRONIKI
I TECHNIK INFORMACYJNYCH

przedmiot

Wprowadzenie do przetwarzania
języka naturalnego - projekt

Transliteracja w tłumaczeniu maszynowym

Projekt końcowy

Bartosz Cywiński, Łukasz Staniszewski

Numer albumu 304025, 304098

prowadzący

mgr inż. Mateusz Klimaszewski

WARSZAWA 31 maja 2022

Spis treści

1. Opis projektu

Celem projektu jest zbadanie, czy transliteracja ma wpływ na efektywność tłumaczenia maszynowego w przetwarzaniu języka naturalnego. W tym celu wykonane zostało porównanie efektywności tłumaczenia zdań arabskich na angielskie do tłumaczenia tych samych zdań arabskich, ale poddanych wcześniej romanizacji, również na angielski. Ze względu na specjalistyczność słownictwa wykorzystanego w projekcie zamieszczone zostały wyjaśnienia niektórych pojęć:

1. **transliteracja** - konwersja tekstu zapisanego oryginalnie znakami jednego alfabetu z użyciem znaków innego alfabetu, oparta na zasadzie ścisłej odpowiedniości liter (zawsze konkretnemu, jednemu grafemowi wejściowego systemu odpowiada jeden, ten sam grafem z drugiego wyjściowego);
2. **romanizacja / latynizacja** - proces transliteracji zakładający, że wyjściowym systemem językowym w procesie jest system opisany przy użyciu alfabetu łacińskiego;
3. **korpus** - zbiór tekstów służący do wykonywania badań lingwistycznych i wspomagający metody uczenia maszynowego (w przypadku tego projektu - do tłumaczenia maszynowego);

2. Wybrane narzędzia

Użytym w projekcie językiem programowania (do przetworzenia danych i liczenia metryk) jest Python wraz ze skryptami Bash. Ponadto wykorzystane zostały narzędzia (z dostępnych publicznie bibliotek):

1. Preprocessing: **SentencePiece** (tokenizacja), **ALA-LC Romanizator** (romanizacja), **langid** (wykrywanie języka).
2. Wykorzystanie i trening modeli - **Fairseq**.
3. Ocena tłumaczenia: **SacreBleu**, **COMET**.

3. Źródło danych

3.1. Dane treningowe

Aby modele nie nauczyły się tylko specyficznego stylu pisania, użyte zostały dane z trzech źródeł (wszystkie korpusy są dostępne za darmo na stronie OPUS). Poniżej przedstawione zostały korpusy, które zostały wykorzystane w projekcie:

1. **CCMatrix** - korpus skonstruowany przez Facebook'a, opisany w [1], składający się z miliardów par zdań w różnych językach. Wartość BLEU dla par zdań arabsko-angielskich mierzona na zbiorze danych składającym się ze zdań wygłaszanych podczas "TED talków"[2] wynosi 28.7. W projekcie wykorzystano milion par zdań arabsko-angielskich pochodzących z tego zbioru.

2. **OpenSubtitles2016** - korpus składający się z napisów do filmów i seriali opisany w [3]. W projekcie użyto 2 miliony par zdań z wersji korpusu z 2016 roku. Dla par zdań arabsko-angielskich wartość BLEU wynosi 25.34.
3. **News-Commentary v16** - korpus składający się z wpisów informacyjnych z amerykańskich mediów [4]. W projekcie wykorzystano wszystkie dostępne pary zdań arabsko-angielskie z tego korpusu, ponieważ są one wysokiej jakości, a także jest ich względnie niedużo, bo tylko nieco ponad 83 000 .

3.2. Dane testowe

Po analizie dostępnej literatury, zdecydowano, że jako zbiór testowy w projekcie wykorzystany zostanie korpus **TED2020** [5] składający się z transkrypcji popularnych TED-talków - jest to zbiór o wysokiej jakości, więc zastosowanie go do ewaluacji jest miarodajne i pozwala porównać modele z dostępną literaturą [1].

4. Przygotowanie danych

4.1. Czyszczenie danych i podział

Zdania z wykorzystywanych korpusach są dostarczone w znacznej większości w odpowiedniej formie (np. bez nadmiarowych znaków). Żeby jednak wyeliminować ewentualny szum w danych odrzucone zostały zdania składające się z mniej niż 3 słów, a także składające się z więcej niż 100 słów. Ponadto wyeliminowane zostały nadmiarowe białe znaki i tagi html. Aby upewnić się, że na wejście modelu trafiają tylko zdania w interesujących językach, przy użyciu pakietu langid wykryte zostały i odrzucone zdania w językach odpowiednio innym od arabskiego i angielskiego. Tak wyczyszczone dane tworzą ostateczne zbiory (nie uwzględniając jeszcze na tym etapie procesu romanizacji). Ostatnim działaniem w tym etapie było wydzielenie ze zbioru treningowego, zbioru walidacyjnego o rozmiarze 10% rozmiaru całego zbioru.

4.2. Romanizacja

Wszystkie przetworzone zdania arabskie zostały poddane transliteracji (przy użyciu narzędzia wspomnianego wcześniej). W wyniku zastosowania konwersji utworzone zostały dodatkowe zbiory danych, na których model będzie zarówno uczony, walidowany jak i testowany. Każde zdanie arabskie znajdujące się w zbiorach dostarczane jest na wejście modelu zaimplementowanego w wykorzystanej bibliotece. Predykcje tego modelu transliterującego (MLE Simple) opierają się na metodzie największej wiarygodności. Model został wytrenowany na wszystkich danych treningowych zawartych w repozytorium biblioteki. Jak przedstawiono w [6] wykorzystany model na zbiorze testowym użytym przez autorów publikacji osiąga skuteczność na poziomie 92.2%, gdzie w procesie ewaluacji transliteracji nie jest uwzględniana wielkość liter ani interpunkcja.

4.3. Tokenizacja

Po zdobyciu wszystkich potrzebnych danych (i ich przetworzeniu), kolejnym etapem była tokenizacja wszystkich zdań (podzielenie zdań na 'wyrazy'). W tym celu dla każdej pary biorącej udział w badaniach: arabski-angielski i arabski po romanizacji-angielski wykonana została następująca sekwencja operacji:

1. Wytrenowanie modelu tokenizacji na zbiorze trenującym.
2. Zakodowanie zdań dla zbioru treningowego, walidacyjnego i testowego.

Dla obu par systemów jęzkowych zdecydowano się na słowniki o rozmiarach 16000, wybrano algorytm kodowania **BPE (Byte Pair Encoding)**, a pokrycie znaków wybrano na poziomie 0.995 - takie ustawienia pozwoliły na pozbycie się ze słownika znaków niepożądanych takich jak emotki czy pojedyncze znaki z alfabetów innych języków (np. chińskiego), a także na zareprezentowanie bogatego w litery alfabetu arabskiego.

4.4. Binaryzacja

Ostatnim elementem niezbędnym do tego, aby można było wykorzystać otrzymane dane, jest ich binaryzacja wraz ze słownikiem otrzymanym w ramach tokenizacji - jest to działanie, którego wymaga użyte w projekcie narzędzie do tłumaczenia maszynowego.

5. Modele i trening

Po zdobyciu przetworzonych danych, eksperymenty rozpoczęto od wytrenowania dwóch modeli tłumaczenia maszynowego - bazowego (dokonującego tłumaczenia z języka arabaskiego na język angielski) oraz poddanego transliteracji (dokonującego tłumaczenia z języka arabskiego zapisanego w alfabecie łacińskim na język angielski). Eksperymenty te wykonano przy użyciu dwóch architektur Sieci Neuronowych: Koder-Dekoder LSTM z mechanizmem Atencji oraz Tiny Transformer.

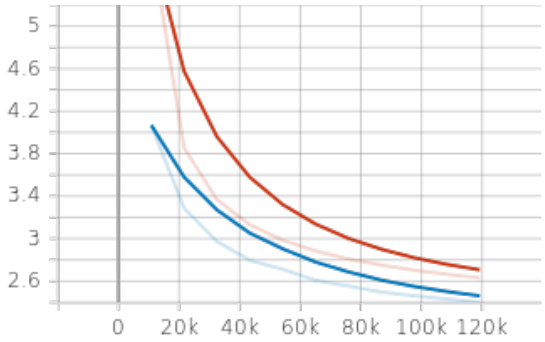
Tabela 5.1. Hiperparametry dla architektur Tiny Transformer

Parametr	Wartość
Liczba epok	24
Learning rate	0.15
Optymalizator	NAG
Dropout	0.2
Seed	0

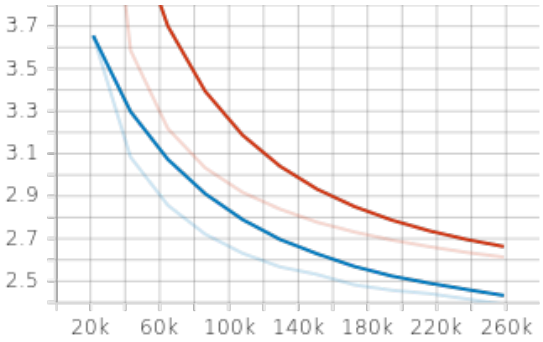
Tabela 5.2. Hiperparametry dla architektur LSTM

Parametr	Wartość
Liczba epok	10
Learning rate	0.15
Optymalizator	NAG
Dropout	0.2
Seed	0

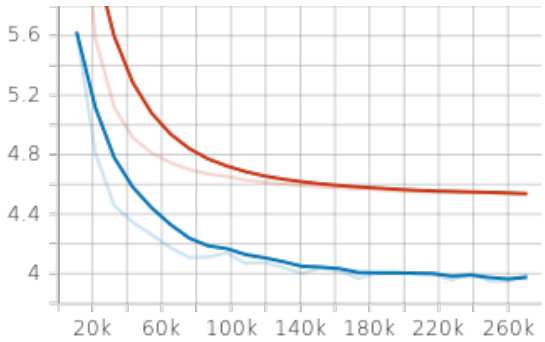
W trakcie uczenia każdego z modeli przy użyciu każdej z architektur, spadek straty prezentował się w następujący sposób, gdzie na osi poziomej zaznaczono liczbę kroków, a na pionowej stratę (UWAGA - ze względu na fakt, że modele bez transliteracji były uczone przy użyciu dwukrotnie większych batch'y, badania z nimi związane mają dwukrotnie mniejszą liczbę kroków):



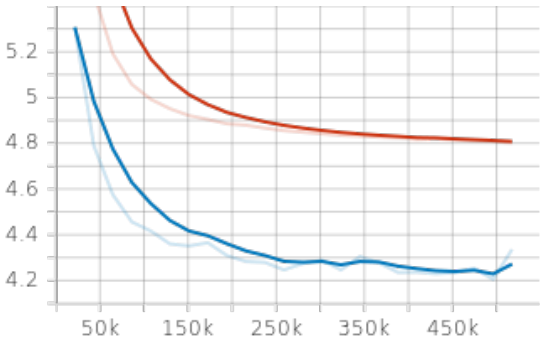
(A) Model bez transliteracji - LSTM



(B) Model z transliteracją - LSTM



(C) Model bez trainsliteracji - Transformer



(D) Model z transliteracją - Transformer

RYSUNEK 5.1. Strata obserwowana w trakcie uczenia każdego z modeli przy użyciu każdej architektury, gdzie kolorem pomarańczowym zaznaczona jest strata na zbiorze treningowym, a niebieskim - na zbiorze walidacyjnym.

6. Ewaluacja

Po wytrenowaniu obu modeli, zostały one ze sobą porównane pod względem skuteczności. W tym celu skorzystano z trzech podstawowych automatycznych metryk do tłumaczenia maszynowego: chrF (character F-score) [7], BLEU [8] oraz COMET [9].

	BLEU	chrF	COMET
LSTM bez transliteracji	**21.45**	**45.14**	**-0.02**
LSTM z transliteracją	21.15	44.50	-0.35
Transformer bez transliteracji	12.01	33.41	-0.54
Transformer z transliteracją	9.87	31.08	-0.85

TABELA 6.1. Zmierzone wartości metryk dla każdego przeprowadzonego treningu.

Wyraźnie widać, że w obydwu przypadkach Transformer poradził sobie znacznie gorzej od LSTM. Patrząc na wykresy funkcji straty można przypuszczać, że dłuższy trening LSTM prowadziłby do uzyskania jeszcze lepszych wyników, ponieważ krzywa uczenia nie zdążyła się jeszcze do końca wypłaszczyć. Natomiast krzywa uczenia Transformera wypłaszczyła się już po około 12 epokach.

Porównując oba modele - ten trenowany na oryginalnym zbiorze danych z tym trenowanym na zbiorze poddanym transliteracji - otrzymano lepsze wyniki dla danych oryginalnych pod względem każdej z metryk. Po metrykach widać bardzo małą różnicę przy okazji LSTM, różnica uwydatnia się tutaj dopiero przy metryce COMET, niemniej jednak w wynikach otrzymanych dla Transformera różnica zaczyna się nieco uwydatniać.

6.1. Analiza predykcji modeli

Z uwagi na to, że dużo lepsze wyniki osiągnął model LSTM, wnioski i analizy prowadzone zostaną na podstawie predykcji tylko z tego modelu.

Po analizie predykcji modeli na zbiorze testowym widać, że dobrze radzą one sobie z tłumaczniem krótkich wypowiedzi, które składają się maksymalnie z ok. 8 słów.

oryginalne zdanie	I see only specific pictures.
bez transliteracji	I see only specific pictures.
z transliteracją	I see only specific pictures.

TABELA 6.2. Oba modele dobrze radzą sobie z tłumaczeniem krótkich zdań.

Na podstawie krótkich zdań trudno zauważyć różnicę między modelem uczonym na danych oryginalnych oraz tym poddanym transliteracji, gdyż predykcje często dokładnie pokrywają się. Jednakże, zbiór testowy składa się również z kilkuzdaniowych wypowiedzi, na których model radzi sobie znacznie gorzej. Na przykładzie dłuższych zdań jest już dostrzegalna różnica w zależności od tego na jakich danych trenowany był model.

oryginalne zdanie	And I had a bit of a wake-up call in Amsterdam: I was there going into the design stores, and mixing with our crowd of designers, and I recognized that a whole lot of stuff pretty much looked the same, and the effect of globalization has had that in our community also.
bez transliteracji	I've got a call from sleep in Amsterdam, I've been there going to a designer shop because combined with a crowd of designers, I have realized that many things look like some of them, and that the impact of globalization has appeared in our society, too.
z transliteracją	I'm going to sleep in Amsterdam, and the effect of globalization has also emerged in our society.

TABELA 6.3. Modele gorzej radzą sobie z dłuższymi zdaniami. Widać znaczną różnicę między tłumaczeniami modeli.

W przedstawionym przypadku w tabeli 6.3 model z transliteracją widocznie pomija tłumaczenie środkowej części oryginalnego zdania, podczas gdy model uczony na oryginalnych danych stara się przynajmniej przetłumaczyć zdanie. Przedstawione powyżej tłumaczenie modelu uczonym na danych bez transliteracji całkiem dobrze oddaje sens zdania, zawiera jednak wiele błędów gramatycznych. Zauważalna jest też zamiana słów w tłumaczeniu na synomimy słów użytych oryginalnie, co zostało przedstawione w tabeli 6.4. Jest to częsty przypadek w predykcjach obu modeli.

oryginalne zdanie	That's a lovely idea.
bez transliteracji	That's a nice idea.
z transliteracją	That's a beautiful idea.

TABELA 6.4. Modele zmieniają słowa w zdaniach na synonimy.

Ciekawym porównaniem predykcji jest też ten przedstawiony poniżej w tabeli 6.5. Można tu zauważyć, że mimo praktycznie identycznych predykcji obu modeli inaczej przedstawiona jest godzina - w jednym przypadku jest ona liczbą a w drugim słowem. W zasadzie ciężko jednoznacznie określić, które z tłumaczeń jest poprawniejsze. Z jednej strony model bez transliteracji w swojej predykcji dał słowo, tak samo jak było to w zdaniu oryginalnym. Z drugiej strony model ten niejako pomija człon ''o'clock''stanowiący o równej godzinie, a model z transliteracją wyraża to dokładnie w postaci liczby.

oryginalne zdanie	They worked until four o'clock in the morning.
bez transliteracji	Keep working till four in the morning.
z transliteracją	Keep working till 4:00 in the morning.

TABELA 6.5. Modele w różny sposób tłumaczą godzinę.

Warto również zwrócić uwagę na to jak modele radzą sobie ze zdaniami podrzędnymi w zdaniach. Przykładowo, gramatycznie wtrącenie w zdaniu jest łatwiejsze do uzyskania przez model nie poddany transliteracji co pokazano w tabeli 6.6.

oryginalne zdanie	Now it's 0.5 – even worse than that in America – showing us the income inequality.
bez transliteracji	Now 0.5 – even worse than that in America – we're building inequality in income.
z transliteracją	Now 0.5-even it is worse in America — we have not equal income inequality.

TABELA 6.6. Modele w różny sposób tłumaczą godzinę.

7. Współdzielenie zagnieżdżeń

7.1. Przygotowanie danych

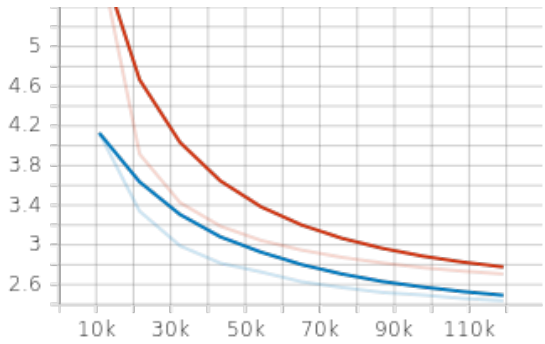
Dodatkowym badaniem przeprowadzonym w ramach projektu było porównanie dotychczas wytrenowanych modeli - LSTM w wersji podstawowej i LSTM poddane transliteracji - z tymi samymi modelami współdzielącymi zagnieżdżenia.

Pierwszym działaniem w tym etapie było ponowne zbinaryzowanie danych tak, aby zbiory wejściowe i wyjściowe (w tym przypadku dwa różne systemy językowe) współdzieliły ten sam słownik, przez co też operowały na słownikach o tej samej wielkości. Dzięki takiemu działaniu możliwe było wykonanie treningu modeli ze współdzielonymi zagnieżdżeniami.

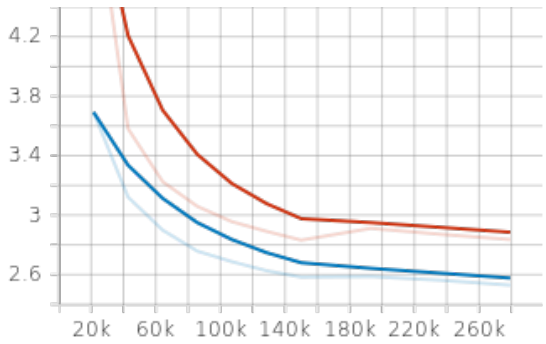
7.2. Trening

Po zdobyciu odpowiednich danych, oba modele (z architekturą LSTM - Autokoder z Atencją - dla modelu bazowego i dla modelu poddanego transliteracji) zostały wytrenowane na tych samych hiperparametrach jak te opisane w tabeli 5.2. Takie podejście umożliwiło 'lepsze'zbadanie wpływu współdzielenia zagnieżdżeń na tłumaczenie maszynowe.

W trakcie uczenia każdego z modeli, spadek straty prezentował się w sposób przedstawiony niżej, gdzie na osi poziomej zaznaczono liczbę kroków, a na pionowej stratę. Tak samo jak poprzednio, trening dla modelu bez transliteracji został wykonany na dwukrotnie większym rozmiarze batch'a, stąd też liczba kroków na wykresie jest dwukrotnie mniejsza.



(A) Model bez transliteracji ze współdzielonymi zagnieżdżeniami - LSTM



(B) Model z transliteracją ze współdzielonymi zagnieżdżeniami - LSTM

RYSUNEK 7.1. Strata obserwowana w trakcie uczenia każdego z modeli ze współdzielonymi zagnieżdżeniami, gdzie kolorem pomarańczowym zaznaczona jest strata na zbiorze treningowym, a niebieskim - na zbiorze walidacyjnym.

7.3. Analiza wyników

Po wytrenowaniu modeli, zostały one porównane na tle modeli bez współdzielenia zagnieżdżeń względem skuteczności. Tak jak poprzednio, skorzystano z automatycznych metryk: chrF (character F-score) [7], BLEU [8] oraz COMET [9].

	BLEU	chrF	COMET
Model LSTM bez współdzielenia zagnieżdżeń, bez transliteracji	**21.45**	45.14	**-0.02**
Model LSTM bez współdzielenia zagnieżdżeń, z transliteracją	21.15	44.50	-0.35
Model LSTM ze współdzielonymi zagnieżdżeniami, bez transliteracji	21.41	**45.18**	**-0.02**
Model LSTM ze współdzielonymi zagnieżdżeniami, z transliteracją	20.62	44.15	-0.36

TABELA 7.1. Zmierzone wartości metryk dla modeli ze współdzielonymi zagnieżdżeniami na tle tych samych, bez współdzielonych zagnieżdżeń.

W każdym przypadku wyniki otrzymane dla modeli ze wpółdzielonymi zagnieżdżeniami nie dały lepszych rezultatów niż modele bez współdzielenia zagnieżdżeń. Nie są to jednak znaczące różnice. Niemniej jednak zabieg współdzielenia zagnieżdżeń w naszym przypadku nie pomógł w uzyskaniu lepszych wyników tłumaczenia.

Przeprowadzenie tego eksperymentu miało potencjał w przypadku modelu z transliteracją, ze względu na możliwość łączenia słowników języków korzystających z tych samych alfabetów, co mogłoby ułatwić uczenie się modeli. Współdzielenie zagnieżdżeń mogłoby zmniejszyć złożoność modelu ze względu na odrzucenie wag modelu związanych z mapowaniami zagnieżdżeń jednego języka do drugiego.

8. Wnioski i podsumowanie

Jedną z hipotez dlaczego otrzymane zostały lepsze wyniki dla oryginalnych danych jest to, że w procesie transliteracji tracona jest nieco dokładność - w końcu użyte w tym projekcie narzędzie osiąga 90% skuteczności. Transliteracja nieidealnym narzędziem może powodować stratę informacji, co może mieć wpływ na różnicę w wynikach obu modeli, skoro drugi z nich korzysta z oryginalnych danych.

Wytrenowane w ramach projektu modele charakteryzują się tym, że potrafią wydobyć sens z tłumaczonego zdania, natomiast przy okazji wyniki translacji są bogate w liczne błędy gramatyczne. Wśród predykcji można zauważyć również zdolność modeli do upraszczania zdań i słów, szczególnie w przypadku przymiotników, gdzie modele preferują przykładowo słowo straszny od słów przerażający i koszmarny. W trakcie analizowania wyników zauważono liczne błędy w predykcjach związane z zaimkami - jest to najprawdopodobniej związane z charakterystyką języka arabskiego, gdzie często w zdaniach pomijany jest podmiot.

W [1], gdzie opisano konstruowanie korpusu CCMatrix, dla ewaluacji powstałego zbioru wytrenowano system tłumaczenia maszynowego na powstałym zbiorze, a następnie dokonano ewaluacji na tym samym zbiorze testowym, który został użyty w naszym projekcie. Dlatego też, wyniki otrzymane w artykule mogą posłużyć jako wyniki referencyjne w tym projekcie. Jak już wcześniej zostało to wspomniane, w referencyjnym rozwiązaniu zmierzona metryka BLEU wyniosła 28.7. Najlepszą wartość metryki BLEU w tym projekcie wystąpiła dla LSTM uczonego na zbiorze bez transliteracji i wyniosła ona 21.45. Zważając na to, że rozwiązanie referencyjne trenowane było na zbiorze 6.5 miliona zdań (ponad dwa razy większym od naszego), a także wybrana w tym projekcie architektura sieci neuronowej - Autokoder z Atencją - nie stanowi state-of-art w tłumaczeniu maszynowym, wyniki można uznać za zadowalające.

Bibliografia

[1] S. Holger, W. Guillaume, E. Sergey, G. Edouard i J. Armand, "CCMatrix: Mining Billions of High-Quality Parallel Sentences on the WEB", 2020.

[2] Q. Ye, S. Devendra, F. Matthieu, P. Sarguna i N. Graham, "When and Why are pre-trained word embeddings useful for Neural Machine Translation", w *HLT-NAACL*, 2018.

[3] L. Pierre i T. Jorg, "OpenSubtitles2016: Extracting Large Parallel Corpora from Movie and TV Subtitles", 2016.

[4] J. Tiedemann, "Parallel Data, Tools and Interfaces in OPUS", angielski, w *Proceedings of the Eight International Conference on Language Resources and Evaluation (LREC'12)*, N. C. (Chair), K. Choukri, T. Declerck i in., red., Istanbul, Turkey: European Language Resources Association (ELRA), maj 2012, ISBN: 978-2-9517408-7-7.

[5] N. Reimers i I. Gurevych, "Making Monolingual Sentence Embeddings Multilingual using Knowledge Distillation", w *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing*, Association for Computational Linguistics, list. 2020. adr.: https://arxiv.org/abs/2004.09813.

[6] E. Fadhl i H. Nizar, "Automatic Romanization of Arabic Bibliographic Records", *arXiv preprint arXiv:2103.07199*, 2021.

[7] M. Popović, "chrF: character n-gram F-score for automatic MT evaluation", w *Proceedings of the Tenth Workshop on Statistical Machine Translation*, Lisbon, Portugal: Association for Computational Linguistics, wrz. 2015, s. 392–395. DOI: 10.18653/v1/W15-3049. adr.: https://aclanthology.org/W15-3049.

[8] K. Papineni, S. Roukos, T. Ward i W.-J. Zhu, "BLEU: A Method for Automatic Evaluation of Machine Translation", w *Proceedings of the 40th Annual Meeting on Association for Computational Linguistics*, ser. ACL '02, Philadelphia, Pennsylvania: Association for Computational Linguistics, 2002, s. 311–318. DOI: 10.3115/1073083.1073135. adr.: https://doi.org/10.3115/1073083.1073135.

[9] R. Rei, C. Stewart, A. C. Farinha i A. Lavie, *COMET: A Neural Framework for MT Evaluation*, 2020. DOI: 10.48550/ARXIV.2009.09025. adr.: https://arxiv.org/abs/2009.09025.