STUDIA SOCJOLOGICZNE 2022
2 (245), 137–164
ISSN 0039−3371, e-ISSN 2545–2770
DOI: 10.24425/sts.2022.141426

Paweł Matuszewski
Collegium Civitas

„ŚMIECI NA WEJŚCIU, ŚMIECI NA WYJŚCIU". WPŁYW JAKOŚCI KODERÓW NA DZIAŁANIE SIECI NEURONOWEJ KLASYFIKUJĄCEJ WYPOWIEDZI W MEDIACH SPOŁECZNOŚCIOWYCH

Jedna z głównych decyzji przy ręcznym kodowaniu danych tekstowych dotyczy tego, czy kodowanie ma być weryfikowane. W przypadku modeli nadzorowanych prowadzi to do istotnego dylematu: czy lepszym rozwiązaniem jest dostarczenie modelowi dużej liczby przypadków, na których będzie się uczyć kosztem weryfikacji poprawności danych, czy też zakodowanie każdego przypadku n-razy, co pozwoli porównać kody i sprawdzić ich poprawność, ale jednocześnie n-krotnie zmniejszy zbiór danych treningowych. Taka decyzja może zaważyć nie tylko na ostatecznych wynikach klasyfikatora. Z punktu widzenia badaczy jest istotna również dlatego, że – realistycznie zakładając, że badania mają ograniczone źródło finansowania – nie można jej cofnąć. Wykorzystując 100 tys. unikatowych i ręcznie zakodowanych tweetów przeprowadzono symulacje wyników klasyfikatora w zależności od kontrolowanego odsetka błędnie zakodowanych dokumentów. Na podstawie danych przedstawiono rekomendacje.

Słowa kluczowe: sieci neuronowe; klasyfikacja danych tekstowych; modele nadzorowane; opinion mining; jakość koderów

„Garbage in, Garbage out". The Impact of Coders' Quality on the Neural Network Classifying Text on Social Media

One of the critical decisions when manually coding text data is whether to verify the coders' work. In the case of supervised models, this leads to a significant dilemma: is it better to provide the model with a large number of cases on which it will learn at the expense of verifying the correctness of the data, or whether it is better to code each case n-times, which will allow to compare the codes and check their correctness but at the same time will reduce the training dataset by n-fold. Such a decision not only affect the final results of the classifier. From the researchers' point of view, it is also crucial because, realistically assuming that research has limited funding, it cannot be undone. The study uses a simulation approach and provides conclusions and recommendations based on 100,000 unique and hand-coded tweets.

Key words: text classification; neural networks; supervised models; opinion mining; quality of coders

Collegium Civitas, pawel.m.matuszewski@gmail.com, ORCID 0000-0003-0069-157X.

Źródło finansowania: grant NCN 2019/03/X/HS6/00882 (Miniatura 3).

Wstęp

Potencjał wykorzystania danych internetowych w badaniach socjologicznych obecnie wydaje się niekwestionowany (Lazer i in. 2009). Na gruncie socjologii polskiej jedne z pierwszych artykułów dostrzegających znaczenie wówczas jeszcze względnie nowego medium pojawiały się na łamach „Studiów Socjologicznych" (Batorski, Olcoń-Kubicka 2006; Batorski, Olechnicki 2007). Z kolei publikacje polskich autorów z ostatnich kilku lat podkreślają rolę wielkich zbiorów danych w analizach społecznych (Jemielniak 2019; Turner, Zieliński, Słomczyński 2018; Żulicki 2017) i wskazują wręcz na nieuchronność uwzględniania w projektach badawczych badania społeczności internetowych (Jemielniak 2018). Poniższy tekst wpisuje się w ten dyskurs poruszając problematykę wykorzystania w socjologii dużych zbiorów tekstowych.

Wzrost dostępnej mocy obliczeniowej, rozwój metod przetwarzania języka naturalnego, postępująca demokratyzacja dostępu do sztucznej inteligencji i związane z nią malejące wymagania techniczne niezbędne do posługiwania się nawet najbardziej zaawansowanymi rozwiązaniami sprawiły, że socjolodzy uzyskali narzędzia do badania olbrzymich zbiorów danych tekstowych, których analiza wcześniej przekraczała możliwości jakiegokolwiek zespołu badawczego (DiMaggio 2015; Grimmer, Stewart 2013; Ignatow 2016; Murthy, Bowman 2014; Salganik 2017). Jednakże nawet oprogramowanie, które krok po kroku prowadzi użytkownika do stworzenia algorytmu klasyfikującego wielkie zbiory tekstu, nie zastąpi badacza przy podejmowaniu istotnych – choć często subiektywnych – decyzji. Wyróżnić można dwa rodzaje takich subiektywizmów. Jeden dotyczy kodowania i interpretacji danych, drugi – wyboru tematu badania i podejmowanych kroków analitycznych (Goldenstein, Poschmann 2019; Nelson 2019). W przypadku uczenia maszynowego to badacz decyduje, co jest jednostką analizy (np. zdanie, wypowiedź, akapit), jak też, w jaki sposób dane tekstowe są wstępnie przygotowywane (np. czy usunięte zostaną znaki interpunkcyjne lub zostanie przeprowadzona lematyzacja). Dodatkowo, w przypadku gdy model ma się uczyć na danych zakodowanych wstępnie przez człowieka, to tylko i wyłącznie badacz decyduje, jakie to będą kategorie i jak dokładnie w ich ramach będzie porządkowany tekst.

Stworzenie dużego zbioru ręcznie zakodowanych danych najczęściej wiąże się z koniecznością utworzenia zespołu koderów. Przy realistycznym założeniu, że fundusze, którymi dysponuje kierownik zespołu, są ograniczone, pojawia się również podstawowy dylemat dotyczący procedury kodowania. Z założenia dane, na których model się uczy, powinny spełniać dwa podstawowe warunki: powinny być wysokiej jakości (nie zawierają błędnie nadanych kodów) i każda klasa powinna dostać wystarczającą ilość materiału (Di Franco, Santurro 2020). Poprzez kontrprzykład można powiedzieć, że pożądanych efektów nie uzyska

się dostarczając 10 perfekcyjnie przygotowanych przypadków, jak też miliona oznaczonych w sposób losowy.

Przypuśćmy, że dostępne środki pozwalają na zakodowanie 20 tys. przypadków. To badacz decyduje, w jaki sposób środki te będą wydatkowane. Pierwsze rozwiązanie polega na tym, aby każdy dokument został zakodowany raz. Zbiór danych zatem będzie możliwie największy, ale nie wiadomo, jaki odsetek przypadków będzie niepoprawnie zakodowany. W konsekwencji model będzie uczył się na maksymalnie dużym, ale przynajmniej w części na nieprawidłowo przygotowanym zbiorze. Inne rozwiązania polegają na tym, aby sprawdzić każdy przypadek n-krotnie i upewnić się, że został opatrzony odpowiednim kodem. Tym samym otrzyma się zbiór zweryfikowanych danych, lecz liczący 20000/n przypadków. Im większa będzie kontrola (np. sprawdzenie każdego przypadku dwu-, trzy- lub czterokrotnie), tym mniejsza końcowa wielkość zbioru. W konsekwencji, liczebność przypadków w poszczególnych klasach może być niewystarczająca, aby uzyskać pożądane wyniki działania klasyfikatora (Hopkins, King 2010).

Pablo Barberá i współpracownicy (Barbera i in. 2021) na podstawie analiz przeprowadzonych na artykułach prasowych wskazują, że najlepszą strategią jest dążenie do osiągnięcia jak największej ilości danych treningowych, kosztem ich weryfikacji. Każdy unikatowy przypadek powinien zatem być zakodowany tylko raz. Argument, który za tym stoi, jest dość prosty. Zakładając, że współpracujemy z doświadczonymi koderami, którzy są ze sobą zgodni, nic nie zyskujemy na kodowaniu n-krotnie tego samego przypadku. Taka procedura nie daje żadnego zysku, bo nie dostarcza żadnej nowej informacji. W rezultacie baza treningowa nie zyskuje na jakości, za to staje się n-razy mniejsza.

Argument Barbery i in., choć logicznie przekonujący, opiera się na założeniu, że koderzy nie popełniają błędów. Oczywiście, można przypuszczać, że w sytuacji gdy sam schemat kodowania jest relatywnie prosty, doświadczony koder, dobrze zaznajomiony z książką kodową i rozumiejący instrukcje kodowania będzie popełniał prawdopodobnie mniej błędów niż osoba, która podejmuje się identycznego zadania po raz pierwszy. Niemniej jednak warto mieć na uwadze, że socjologów raczej interesują klasyfikacje nawiązujące do teorii socjologicznych, nie zaś proste podziały. Im bardziej skomplikowany schemat kodowania, tj. więcej kategorii i większa potrzeba interpretacji znaczenia zawartego w tekście, tym większe ryzyko rozbieżności między koderami, a także kodami nadawanymi przez tego samego kodera. Do błędów wynikających z interpretacji dochodzą błędy wynikające z omyłkowego przypisania kodu niezgodnie z intencją, rozkojarzenia, zmęczenia itd. Jeśli zatem można oczekiwać po doświadczonych koderach lepiej wykonanego zadania, to wątpliwe jest uznanie, że są nieomylni. To prowadzi do głównego pytania badawczego tego artykułu: *Jaki odsetek nieprawidłowo zakodowanych przypadków w zbiorze treningowym uzasadnia*

procedurę jednokrotnego kodowania? Inaczej mówiąc, jak bardzo zgodni muszą być koderzy, aby zwiększona liczba danych rekompensowała błędy wynikające z nieodpowiednio nadanych kodów. Odpowiedź na to pytanie, a także opracowanie rekomendacji wymagają zadania pytań pomocniczych. Obejmują one ustalenie relacji między odsetkiem nieprawidłowo zakodowanych przypadków a działaniem klasyfikatora oraz czy suboptymalne działanie jest możliwe do wykrycia, gdy nieweryfikowane są nie tylko dane treningowe, ale i dane testowe. Kwestia ta nie została do tej pory szczegółowo opisana w literaturze. Dotychczasowe nieliczne analizy podejmowane były zaś na małych próbach i relatywnie prostych zadaniach klasyfikacyjnych (He, Schonlau 2020b, 2020a).

Udzielenie odpowiedzi na pytanie badawcze wymaga porównania, jak klasyfikator sobie radzi w sytuacji, gdy kontrolowana jest zarówno liczba przypadków, na których trenowany jest model, jak też odsetek nieprawidłowo zakodowanych danych. Należy też zwrócić uwagę na dodatkową komplikację, która utrudnia odpowiedź. W sytuacji, gdy błędy popełniane są losowo (np. koder wpisuje kody w przypadkowej kolejności, aby terminowo wykonać zadanie), można się spodziewać, że końcowy model będzie gorzej działał niż model wytrenowany na poprawnych danych. Oczywiście takie losowe błędy mogą się pojawić (choćby przez rozproszoną uwagę), to w przypadku rzetelnych kodujących częściej spodziewać się można, że będą dokonywać systematycznego błędu opartego na złym zrozumieniu schematu kodowania. W takiej sytuacji kodowanie będzie spójne, ale niezgodne z intencją autora schematu. Wówczas model może nauczyć się takiej nieprawidłowej „interpretacji" i zacząć nadawać konsekwentnie błędny kod. Wtedy, im więcej systematycznie błędnych danych model otrzyma, tym bardziej pewny będzie swoich wyników i tym większe będą rozbieżności między oczekiwanym a rzeczywiście wytrenowanym klasyfikatorem. Problematyczne w tej sytuacji jest to, że prawdopodobnie ewaluacja nie wykaże nieprawidłowości, jeśli ci sami koderzy są odpowiedzialni za przygotowanie zbioru treningowego i testowego, ponieważ dane testowe będą zawierały podobny odsetek źle zinterpretowanych przypadków, co dane treningowe.

Dane i metody[1]

Uczenie maszynowe i sieci neuronowe

Do klasyfikacji tekstu wykorzystuje się dwa rodzaje modeli maszynowych: nienadzorowane i nadzorowane. Pierwsze polegają na tym, że algorytm sam na podstawie dostarczonych mu danych określa, w jaki sposób można podzielić

[1] Dane oraz kod użyty w tym artykule mogą być udostępnione przez autora na uzasadnioną prośbę.

tekst. Z ich pomocą można eksplorować zależności, których istnienie jest niełatwe do przewidzenia (Bail 2014; Grimmer, Stewart 2013). Nie sprawdzą się one jednak, gdy liczba kategorii jest z góry ustalona (jak w analizie sentymentu) lub łatwa do określenia (Lin, He 2009). Algorytm bowiem nie rozpoznaje intencji badacza i porządkuje tekst na podstawie powiązań statystycznych, a nie, na przykład, teorii socjologicznej. W konsekwencji użycie go niezgodne z przeznaczeniem może prowadzić do frustrujących i najczęściej skazanych na porażkę prób pogodzenia wyników z wcześniej sformułowanymi kategoriami analitycznymi. Modele nienadzorowane są popularnym rozwiązaniem wśród badaczy, co może wiązać się z tym, że w odróżnieniu od modeli nadzorowanych nie wymagają wcześniejszego przygotowania kosztownego (pod względem finansowym i/lub włożonego wysiłku) dużego, ręcznie zakodowanego, wysokiej jakości zbioru treningowego (Miller, Linder, Mebane 2020). Są zatem relatywnie bardziej dostępne dla badaczy nieposiadających wsparcia finansowego lub zespołu gotowego podjąć się ręcznego kodowania.

W przypadku, gdy istnieją ustalone kategorie kodowe, właściwym wyborem są modele nadzorowane. Polegają one na znalezieniu argumentów funkcji, która łączy tekst z przypisanym do niego kodem. Im więcej i im lepsze przykłady taki model dostanie, tym lepiej nauczy się klasyfikować tekst. Można go wówczas wykorzystać do tego, aby samodzielnie zakodował dowolną liczbę danych, których wcześniej nie widział (Di Franco, Santurro 2020; Jordan, Mitchell 2015).

Im bardziej skomplikowane dane, tym bardziej złożona funkcja jest potrzebna, aby połączyć tekst z odpowiednią klasą. W najprostszych zadaniach klasyfikacyjnych wystarczająca może się okazać regresja logistyczna. Uwzględniając jednak stopień skomplikowania tekstu i kategorii, które zazwyczaj interesują socjologów, niezbędne do uzyskania satysfakcjonujących wyników może okazać się użycie bardziej zaawansowanych modeli uczenia maszynowego w postaci płytkich – jak w przypadku opisanym w tym artykule – lub głębokich sieci neuronowych (Di Franco, Santurro 2020; Joulin i in. 2016).

Zbiór danych tekstowych

Dane tekstowe pochodziły z platformy Twitter. Choć nie jest to najbardziej popularne medium społecznościowe w Polsce (6,5 mln użytkowników w porównaniu do 21,5 mln użytkowników Facebooka, źródło: badanie Mediapanel, grudzień 2020), to stanowi miejsce dynamicznych dyskusji na tematy związane z polityką i aktualnymi wydarzeniami. Na Twitterze swoje konta posiadają prezydent, wszystkie główne partie polityczne, 85% posłów, 47% senatorów i 89% europarlamentarzystów, którzy łącznie są obserwowani przez ponad 2,5 mln użytkowników (dane własne, stan na 01.08.2021 r.).

Wykorzystane w tym artykule dane były zbierane przez cały 2019 rok (poprzez Twitter REST API; zob. Rodak 2017) i pochodziły z kont, które spełniały

następujące warunki. Po pierwsze, musiały to być konta, które systematycznie (minimum 3 razy w tygodniu) zamieszczają wypowiedzi dotyczące polityki i współczesnych wydarzeń. Dzięki temu możliwe było pobranie próby tweetów istotnych ze względu na temat badań z relatywnie niewielkim odsetkiem treści dla projektu nieważnych (wpisów komentujących pogodę, życzeń miłego weekendu itp.). Po drugie, były to konta, które publikowały wyłącznie lub głównie w języku polskim. Po trzecie, były to konta, które można traktować jako liderów opinii (Cha i in. 2010; Watts, Dodds 2007), tj. obserwowanych przez znaczną liczbę użytkowników (minimum 5000) i takich, których tweety są rozpowszechniane na platformie (mediana udostępnień była powyżej 50). Skupienie się na tej grupie kont miało zwiększyć prawdopodobieństwo, że tweety będą napisane poprawnym językiem, a przekonania będą wyrażone klarownie. Z bazy usunięto retweety, aby ten sam tekst się nie powtarzał, a także cytowania i odpowiedzi, ponieważ do ich zrozumienia często niezbędny jest szerszy kontekst. W efekcie zebrano 1 228 417 unikatowych wypowiedzi, z których za pomocą losowania prostego pobrano próbę w liczbie 100 tys. wpisów. Zakładano, że taka liczebność wystarczy, aby wypełnić sugerowane kryterium dostarczenia co najmniej 100 przypadków na każdy kod (Grimmer, Stewart 2013; Hopkins, King 2010).

Wstępne przetworzenie danych tekstowych

Dane tekstowe, na których uczył się model, zostały wstępnie przetworzone. Procedura obejmowała usunięcie znaków interpunkcyjnych, emotikonów, emoji, zmianę pisowni tak, aby wszystkie słowa zaczynały się małą literą oraz lematyzację, tj. sprowadzenie słów do ich formy podstawowej (np. szliśmy → iść, domów → dom). Celem było znormalizowanie danych, które zazwyczaj pomaga w podobnych zadaniach klasyfikacyjnych (Denny, Spirling 2018; HaCohen-Kerner, Miller, Yigal 2020; Haddi, Liu, Shi 2013). Warto wspomnieć, że w przypadku Twittera tekst jest pisany przez różnych użytkowników, z różnym wykształceniem, czasami w pośpiechu i nie ma na dodatek (przynajmniej w momencie, gdy te dane powstawały) możliwości edycji raz napisanej wiadomości. Z tego względu zarówno reguły pisowni, jak też emotikony i emoji są stosowane i interpretowane indywidualnie w zróżnicowany sposób (Bai i in. 2019; Brants, Sharif, Serebrenik 2019), który może tworzyć dodatkowy szum w danych i utrudniać modelowi uczenie się.

Procedura kodowania danych

Zespół kodujący tweety składał się z siedmiu osób – studentów drugiego stopnia nauk politycznych. Każdy tweet był kodowany dwukrotnie (tj. 100.000 unikatowych przypadków zostało zakodowane dwa razy), a koderzy otrzymywali zapłatę adekwatną do liczby zakodowanych wypowiedzi. Koder A zakodował

50.000 tweetów, koder B 35.000, koder C 25.000, koder D 25.000, koder E 22.500, koder F 22.500 i koder G 20.000. Praca koderów była na bieżąco monitorowana.

Kodowanie materiału jakościowego jest wymagającym zadaniem. Nawet tak krótka wypowiedź jak tweet (maksymalnie 280 znaków) może mieć wiele warstw znaczeniowych, z których każda może podlegać wielu interpretacjom (Ignatow 2016; Monroe 2019). Proste zadania, np. oznaczenie tweeta z nazwiskiem posła, generują stosunkowo niewiele błędów. Ich źródło nie leży w interpretacji, lecz np. zmęczeniu, pośpiechu czy omyłkowym nadaniu błędnego kodu. Jednakże, w przypadku badań socjologicznych, badaczy zazwyczaj interesują bardziej pogłębione dane. Im trudniejszy tekst do interpretacji i mniej precyzyjne instrukcje kodowania, tym bardziej rośnie prawdopodobieństwo pojawiania się błędów (DiMaggio 2015; Grimmer, Stewart 2013). Z tego względu stworzono szczegółową książkę kodową z dokładnymi instrukcjami. W pierwszej fazie, na trzech osobnych spotkaniach, zespół koderów wraz z kierownikiem omówili książkę kodów i zakodowali wspólnie pierwsze 700 tweetów (po ok. 200–250 na spotkanie). Celem było wystandaryzowanie sposobu interpretacji materiału badawczego i nadawania kodów (Krippendorff 2003). W kolejnej fazie koderzy pracowali osobno według indywidualnych grafików. Każdy z nich miał określoną minimalną ilość pracy do wykonania w danym czasie, aby członkowie zespołu równolegle i w zbliżonym tempie rozwijali rozumienie zadania i ewentualnie napotykali zbliżone problemy. Ten etap można traktować jako wypracowywanie i doprecyzowywanie pierwszej wersji książki kodowej. W tym czasie jakiekolwiek wątpliwości były omawiane na forum całego zespołu, a także podejmowana była decyzja o rozszerzaniu jakiejś kategorii bądź tworzeniu nowego kodu. Procedura kodowania była na bieżąco monitorowana. Przypadki rozbieżności w kodowaniu były ponownie sprawdzane (trzeci raz przez niezależnego kodera) i jako prawidłowy wybierany był kod, który się powtarzał. Jeśli trzeci kod był inny od dwóch wcześniejszych, to ostateczną decyzję podejmował kierownik zespołu jako najbardziej doświadczony badacz. Dzięki tej procedurze powstał zbiór zweryfikowanych danych treningowych, jak też możliwy był do wyodrębnienia zbiór danych, które zostały zakodowane niepoprawnie.

Analiza wyników dotyczy rozpoznawania przekonań autorów tweetów na tematy związane z polityką i aktualnymi wydarzeniami. Zadanie to można podzielić na dwie części: identyfikację tego, co jest przedmiotem sądu (o czym mówi autor) i określenie, jaki autor ma do tego stosunek (Ajdukiewicz 1965; Boudon 1997; Marciszewski 1972; Ziółkowski 1989). Przez temat rozumieć zatem należy to, czego dotyczy wypowiedź. Instrukcja dla koderów wskazywała, aby przeformułowywali tweety według schematu: „Autor tweeta uważa, że …". Podmiot zdania podrzędnego wskazywał temat. Tweety jako krótkie formy prawie zawsze były monotematyczne. Z tego względu jednej wypowiedzi przypisywany

był jeden kod. Wyjątkiem były sytuacje, w którym użytkownicy krytykowali partie/polityków poruszając różne wątki. Na takie przypadki została stworzona osobna kategoria. W tabeli 1 są zamieszczone przykładowe instrukcje z książki kodowej.

Tabela 1. Przykładowe instrukcje dla kodujących z książki kodowej

Kategoria	Kod	Opis
Inne	Niezidentyfikowane	Wypowiedź, która jest niezrozumiała lub brakuje kontekstu do jej zrozumienia (np. zdjęcia, filmu, grafiki, artykułu)
	Off	Wypowiedzi nieodnoszące się do żadnych z zarysowanych tematów, np. nastrój osobisty, informowanie o własnych sukcesach, wydarzeniach z życia, informowanie obserwujących o osobistych przemyśleniach na tematy niezwiązane z polityką (np. sport, katastrofy naturalne, pogoda)
Bezpieczeństwo	Bezpieczeństwo	Wypowiedzi odnoszące się do kwestii bezpieczeństwa w Polsce i na świecie, NATO, zagrożenia ze strony innych państw, manewrów i ćwiczeń wojskowych, zakupów uzbrojenia, offsetów, technologii militarnej/wojskowej, stanu polskiej armii
Politycy i polityka	Afera_Dochody	Wypowiedzi dotyczące niejasności w zakresie majątku posiadanego przez polityków, nabycia/zbycia tego majątku, przeniesienia praw majątkowych na bliskich, zeznań majątkowych
	Afera_Nepotyzm	Wypowiedzi dotyczące niejasności w związku z zatrudnianiem osób z kręgów politycznych w administracji państwowej, samorządowej i spółkach Skarbu Państwa (np. Orlen, PZU, PGE)
Kwestie społeczne	Ekologia	Wypowiedzi dotyczące środowiska naturalnego, jego stanu, dewastacji, a także ochrony środowiska naturalnego; wypowiedzi na temat Grety Thunberg, zmian klimatycznych, gazów cieplarnianych, szczytów klimatycznych, OZE

Warto podkreślić, że takie zadanie klasyfikacyjne jest inne niż popularna analiza sentymentu, która jest definiowana najczęściej jako emocjonalny kontekst wypowiedzi (Bakliwal i in. 2013; Drus, Khalid 2019; Mozetič, Grčar, Smailović 2016; Tomanek 2017). Stwarza to istotne problemy między innymi, gdy wydźwięk emocjonalny jest inny niż sąd lub wyrażone jest przekonanie preskryptywne. Przykładem pierwszego przypadku może być zdanie *Cieszę się, że X nie jest już ministrem*, które zdradza krytyczny stosunek autora do X, choć ton wypowiedzi jest pozytywny. Przykładem drugiego jest zdanie *Partia P powinna obniżyć podatki*. W zasadzie trudno określić emocjonalny wydźwięk takiej wypowiedzi i prawdopodobnie większość klasyfikatorów uznałaby, że jej wydźwięk jest neutralny. Można również spekulować, czy kryje się za nią rozczarowanie (*szkoda, że partia P nie obniżyła podatków*), gniew (*dlaczego oni jeszcze nie obniżyli podatków?!*), troska (*dobrze im życzę i gdyby podwyższyli podatki, to mieliby większe poparcie*) itd. Trudno to określić bez szerszego

kontekstu. Zastosowana tutaj metoda omija ten problem, ponieważ przedmiotem wyrażonego sądu są podatki, a takie sformułowanie wskazuje, że według autora wypowiedzi są one za wysokie. Dlatego ta wypowiedź otrzymałaby odpowiednio kod (temat) „podatki" i kod (stanowisko) „negatywny" (por. Mohammad i in. 2016; Sobhani i in. 2019).

Książka kodowa składała się z 65 klas tematów i 3 możliwych stanowisk. Pierwsze 10 najczęściej występujących tematów obejmowało 68% całej bazy danych. Ze względu na charakter analizy zredukowano bazę danych do tematów, które pojawiają się w zbiorze nieprawidłowo zakodowanych wypowiedzi częściej niż 50 razy (35 kategorii). 16% bazy danych stanowią tweety zakodowane jako pozytywne wobec przedmiotu wypowiedzi, 39% jako negatywne i 45% jako neutralne lub ambiwalentne.

Procedura badawcza

W analizach zostały wykorzystane dwa zbiory danych: 1) obejmujący sprawdzone i poprawnie zakodowane tweety; 2) obejmujący niepoprawnie zakodowane przez koderów tweety i podzielony na dwa podzbiory zawierające błędnie wskazane stanowisko (N=8101) i błędnie zidentyfikowany temat (N=11225). Bazy poprawnie i niepoprawnie zakodowanych danych są rozłączne (tj. ten sam tweet nie może się pojawić w obu), aby uniknąć sytuacji, w której ta sama wypowiedź jest zaklasyfikowana na dwa różne sposoby.

W badaniu sprawdzono, w jaki sposób zachowują się sieci neuronowe biorąc pod uwagę procedurę uczenia algorytmu, liczebność danych treningowych, stosunek prawidłowo zakodowanych tweetów do nieprawidłowo zakodowanych i rodzaj zadania klasyfikacyjnego.

W procesie uczenia zastosowano dwie metody. Pierwsza, nazywana standardową, uczeniem biernym, proporcjonalnym lub pasywnym, polegała na tworzeniu zbiorów treningowych na podstawie losowania warstwowego (warstwą były klasy). W konsekwencji próba zachowywała te same proporcje poszczególnych kodów co zbiór, z którego była wylosowana. Oznacza to, że model uczy się danych niezbalansowanych, tj. takich, w których niektóre kategorie są bardzo liczne, podczas gdy inne pojawiają się wyraźnie rzadziej. W konsekwencji może to prowadzić do trudności w rozpoznawaniu najrzadziej występujących kategorii albo dawać złudzenie dobrze działającego algorytmu. Na przykład, w sytuacji gdy jest kilka kategorii, lecz jedna z nich obejmuje 70% danych, wystarczy, aby model nauczył się wszystko rozpoznawać jako tę kategorię, aby zawsze poprawnie klasyfikować w 70% przypadków.

Druga procedura to uczenie aktywne (Chen, Mani, Xu 2012; Miller i in. 2020; Wiedemann 2019). W odróżnieniu od wcześniej opisanego uwzględnia ono informację na temat prawdopodobieństwa, że dany kod został przypisany prawidłowo. Ogólny schemat polega na tym, że w pierwszej fazie model uczy

się na podstawie pierwszej partii zakodowanych danych. W następnej kolejności zamiast np. zlecać koderom zakodowanie kolejnej losowej partii danych, przeprowadza się losowanie danych do zakodowania i wybiera się te, wobec których model ma największe „wątpliwości". Można tego dokonać na różne sposoby, między innymi wykorzystując różne algorytmy do tych samych zbiorów i sprawdzać, w których przypadkach pojawiają się największe rozbieżności lub wykorzystywać prawdopodobieństwa prawidłowej klasyfikacji określane przez algorytm (nie wszystkie algorytmy generują takie dane). Idea stojąca za uczeniem aktywnym polega na tym, aby nie wykorzystywać pracy koderów do zakodowania oczywistych przypadków, z którymi model nie ma problemu. W takiej sytuacji dodanie kolejnych danych będzie miało znikome znaczenie, bo model już z takimi przypadkami sobie radzi. Celem uczenia aktywnego jest dostarczenie zbioru treningowego, na który składać się będą takie dane, z którymi model sobie jeszcze nie radzi. Inaczej mówiąc, procedura polega na intencjonalnym zaburzeniu „losowości" danych wsadowych, tak aby zawierały jak najwięcej przypadków, których klasyfikacja może sprawiać trudności i na których jednocześnie model może się dalej uczyć. Używając szkolnej analogii, można powiedzieć, że nie ma specjalnego uzasadnienia, aby w pracy domowej z matematyki uczeń liceum miał zadania z prostego dodawania, ponieważ na tym poziomie kształcenia ta umiejętność jest już dawno opanowana. Oczywiście zrobi on to bez problemu, ale dla niego to ćwiczenie nie będzie niosło za sobą żadnych korzyści. Jedynie potwierdzi, że posiada dawno opanowaną umiejętność.

Ze względu na różne sposoby uczenia się można spodziewać się odmiennych efektów końcowych. W przypadku uczenia pasywnego zbiór treningowy jest zawsze losowy. W związku z tym im większy odsetek nieprawidłowych danych, tym trudniej ustalić algorytmowi właściwą funkcję, która łączyłaby kod z tekstem. Można się spodziewać, że taka prawidłowość będzie obserwowana niezależnie od wielkości zbioru treningowego. W przypadku uczenia aktywnego wybierane są najtrudniejsze przypadki, ale nie wiadomo, czy model nie jest pewien nadawanych kodów ze względu na złożoność przypadku, czy też dlatego, że bardzo podobne przypadki są kodowane odmiennie na skutek błędu kodowania. W konsekwencji będzie dobierał on tweety w sposób, który dostarczy więcej informacji na temat właściwego sposobu kodowania. Zgodnie z tą logiką działania można spodziewać się, że nauczy się rozpoznawać i „ignorować" niepoprawnie oznaczone tweety szybciej niż przy metodzie pasywnej.

W badaniu pierwsza procedura polegała na losowym wybraniu do zbioru treningowego 1000 tweetów (o określonej proporcji tweetów poprawnie i niepoprawnie zakodowanych) i następnie powiększanie go o kolejny tysiąc (zachowując te same proporcje niepoprawnych kodów), aż zostanie osiągnięty limit danych. Ewaluacja dokonywana jest na losowej próbie 1000 tweetów, które nie są częścią zbioru treningowego. „Sprawdzian" następuje zatem na danych,

których model wcześniej nie widział, co jest zgodne z jego oryginalnym przeznaczeniem (zastąpienie pracy ludzkiej w klasyfikowaniu nowych tweetów).

Druga metoda jest bardziej skomplikowana. Najpierw losowanych jest 1000 tweetów (procedura losowania zbioru treningowego jest taka sama jak w uczeniu pasywnym), na których uczy się model. Następnie z tweetów, które nie znalazły się w zbiorze treningowym, losowany jest zbiór testowy liczący 10000 przypadków. Z tego zbioru uzyskiwane są identyfikatory 1000 tweetów, wobec których prawdopodobieństwo prawidłowej klasyfikacji jest niższe niż 75%. Oznacza to dość wysoki próg pewności, jaki musi spełnić model, aby jego klasyfikacja została uznana za prawidłową. W przypadku, gdy liczba tweetów poniżej progu jest mniejsza niż 1000, to do danych treningowych dodawane są wszystkie tweety. Przypadki są losowane w ustalonej proporcji prawidłowych do nieprawidłowych.

Z opisu metod uczenia można wywnioskować, że jedną ze zmiennych niezależnych jest liczebność danych treningowych. Warunek ten został zasymulowany poprzez wylosowanie pierwszej partii 1000 tweetów oraz dodawanie kolejnych. W przypadku uczenia pasywnego było to 1000 rekordów aż do momentu wyczerpania zbioru nieprawidłowych danych. W przypadku uczenia aktywnego wraz z trenowaniem modelu na większej liczbie danych, suma rekordów, co do których prawdopodobieństwo przypisania prawidłowego kodu wynosiło poniżej 75%, była niższa niż 1000. Z tego względu początkowe interwały (i = 1000) stopniowo malały.

Kolejna zmienna obejmowała proporcje między prawidłowo a nieprawidłowo zakodowanymi tweetami. W symulacjach sprawdzano, jaki wpływ na model ma dostarczenie mu 0%, 10%, 20%, 30%, 40% i 50% niepoprawnie zakodowanych danych. Ewaluacja była dokonywana na dwóch zbiorach testowych, w skład których wchodziły dane wcześniej niewidziane przez model. Pierwszy zbiór zawierał ten sam odsetek niepoprawnie zakodowanych tweetów, co zbiór treningowy. Drugi zbiór zawierał dokładnie te same dane co pierwszy, ale wszystkie z prawidłowymi etykietami. Na tej podstawie możliwe jest sprawdzenie, jaki efekt mają błędnie zakodowane tweety na działanie modelu, jak też czy model wytrenowany na niepoprawnych kodach zachowuje się podobnie jak ten wytrenowany na poprawnych. W tym drugim przypadku testowane jest, czy model rozpoznaje możliwe systematyczne błędy popełniane przez koderów i traktuje jako prawidłowe kody. Jeśli tak się dzieje, to „wewnętrzna" ewaluacja nie powinna się różnić od ewaluacji na poprawnych danych. Jeśli tak nie jest, to pojawia się argument za tym, że model lepiej uczy się na poprawnych danych.

Uwzględnienie wszystkich warunków, tj. zadań (2 zadania), wielkości zbiorów danych treningowych (od 1000 do wyczerpania nieprawidłowo zakodowanych danych w interwałach ok. 1000; w przypadku stanowiska 8000, a w przypadku tematów politycznych 11000), metody uczenia się (2) i odsetka

niepoprawnie nadanych kodów (od 0 do 50%; 6 wartości) w rezultacie stworzyło sieć wielu różnych konfiguracji. Zwłaszcza przy niedużych zbiorach treningowych działanie modelu w dużej mierze może być związane z losowym doborem przypadków, który mogą pokazać przypadkowo odbiegające od normy pod względem stopnia trudności. Aby wyeliminować wpływ losowości każda konfiguracja była powtarzana 50 razy. Liczba rekordów w analizowanym zbiorze danych wynosiła zatem łącznie 11400 [(8 x 2 x 6 + 11 x 2 x 6) x 50].

W badaniu klasyfikacja dotyczyła danych nierównomiernie rozłożonych między klasami, dlatego zdecydowano się użyć miary F1, która jest bardziej odporna na takie przypadki niż również popularna dokładność (*accuracy,* Mozetič et al. 2016; Tharwat 2020). Miara F1 dla klasy A jest średnią harmoniczną stosunku poprawnie rozpoznanych elementów z A do wszystkich, które klasyfikator oznaczył jako A (*precyzja; precision*) i stosunku poprawnie sklasyfikowanych elementów z A do wszystkich elementów klasy A (*czułość; recall*), przy założeniu, że obie te miary są jednakowo ważne:

$$F1=2 \times \frac{(\text{precyzja}+\text{czułość})}{(\text{precyzja} \times \text{czułość})}$$

W przypadku miary F1 dla wszystkich klas wyniki są uśredniane.

Ze względu na cele badawcze miara F1 była liczona z wykorzystaniem różnych danych. Sprawdzenie, jaki efekt ma błędne kodowanie na działanie modelu, wymaga użycia do trenowania kontrolowanego odsetka błędnych danych, zaś do testu – danych zaklasyfikowanych prawidłowo. Z kolei sprawdzenie, czy model potrafi osiągnąć ten sam wynik ewaluacji, jeśli będzie trenowany na błędnie kodowanych danych, oznacza, że dane testowe są w równym stopniu błędnie zakodowane. Odzwierciedla to sytuację, w której badacz posiada bazę danych ręcznie zakodowanych przypadków, ale dzieląc ją na podzbiór treningowy i testowy nie wie, że są one nieprawidłowe.

W badaniu wykorzystano publicznie dostępny algorytm fastText, stworzony przez Facebook Artificial Intelligence Research Lab. Cechuje się on dobrymi wynikami w zadaniach dotyczących klasyfikacji tekstu i szybkością działania (Anon 2021; Joulin i in. 2016). Ta ostatnia cecha miała dość istotne znaczenie, ponieważ umożliwiła przeprowadzenie kilkudziesięciu tysięcy symulacji wyników na domowym komputerze. Biorąc pod uwagę skalę projektu wydłużenie procesu uczenia się o jedną minutę oznaczałoby wydłużenie czasu obliczeń o 18 dni. Warto też wspomnieć, że fastText jako oficjalne narzędzie może być wykorzystywany przez wielu badaczy, w tym także w celu sprawdzenia czyichś wyników. Zaletą jest też to, że jest biblioteką typu open-source i jej dokumentacja została zamieszczona na githubie (https://github.com/facebookresearch/fastText/releases/tag/v0.9.2). Szczegółowy opis architektury algorytmu wykracza jednak

poza zakres tekstu skierowanego do socjologów. Takie informacje można uzyskać, na przykład, w artykule Nishana Subedi (2018) oraz autorów biblioteki (Joulin i in. 2016). Dokumentacja fastText to niewątpliwa przewaga w stosunku do samodzielnie budowanych, słabo udokumentowanych i słabo przetestowanych (z reguły przez nikogo poza ich autorami) sieci neuronowych, które są trudniej replikowalne, dają najwyżej zbliżone rezultaty i są wolniejsze (np. samodzielnie stworzony przez autora model dwukierunkowej długiej pamięci krótkoterminowej – biLSTM – dawał uśredniając nieco gorsze rezultaty i uczył się 30-krotnie wolniej, co właściwie uniemożliwiało przeprowadzenie tak dużej liczby symulacji).

Analiza wyników

W celu pełnej odpowiedzi na pytania badawcze użyto sześciu modeli regresji liniowej, w których zmiennymi zależnymi były:

1) miara F1 dla klasyfikatora stanowiska obliczona na zbiorze testowym zawierającym zweryfikowane przypadki;
2) miara F1 dla klasyfikatora tematu obliczona na zbiorze testowym zawierającym zweryfikowane przypadki;
3) różnica między miarą F1 obliczoną dla klasyfikatora stanowiska na sprawdzonym zbiorze testowym oraz miarą F1 obliczoną na zbiorze testowym zawierającym błędnie zakodowane tweety;
4) różnica między miarą F1 obliczoną dla klasyfikatora tematu na sprawdzonym zbiorze testowym oraz miarą F1 obliczoną na zbiorze testowym zawierającym błędnie zakodowane tweety;
5) różnica między miarami F1 dla modelu stworzonego na n liczbie zweryfikowanych przypadków i 2n niezweryfikowanych przypadków obliczona na podstawie zweryfikowanych danych testowych dla klasyfikatora stanowiska;
6) różnica między miarami F1 dla modelu stworzonego na n liczbie zweryfikowanych przypadków i 2n niezweryfikowanych przypadków obliczona na podstawie zweryfikowanych danych testowych dla klasyfikatora tematu.

We wszystkich modelach główną zmienną niezależną był odsetek poprawnie zakodowanych przypadków. Zmiennymi kontrolnymi była wielkość zbioru treningowego i metoda uczenia algorytmu.

Wniosek 1. Model uczony na nieprawidłowych danych działa poniżej swojego optimum

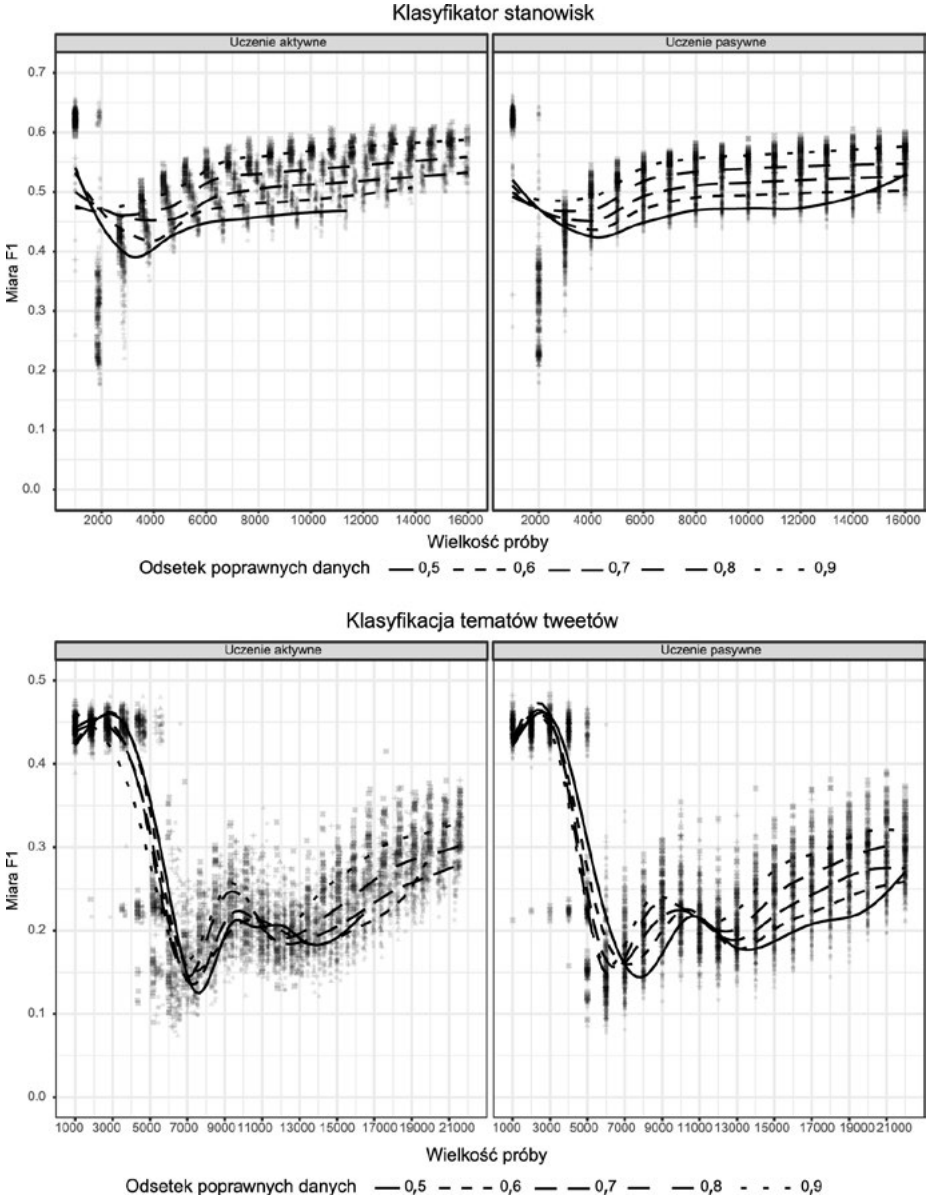
Dane na wykresie 1 sugerują, że w obu zadaniach klasyfikator potrzebuje kilku tysięcy przypadków treningowych, aby jego działanie się ustabilizowało. Poziom ten zdaje się być tym większy, im bardziej skomplikowane jest zadanie, które stoi przed klasyfikatorem.

Wyniki analiz regresji liniowych przedstawione w tabeli 1 wskazują, że wraz ze wzrostem odsetka niepoprawnie zakodowanych tweetów o 10 punktów procentowych miara F1 spada o ok. 0,016–0,026 punktu w przypadku klasyfikacji stanowisk, i 0–0,013 punktu w przypadku klasyfikacji tematów. Takie wyniki oznaczają też, że miara F1 jest w przybliżeniu wprost proporcjonalnie zależna od odsetka błędnie zakodowanych tweetów. Im jest on większy, tym model gorzej klasyfikuje.

Wyniki dla klasyfikatora stanowisk wyrażanych na Twitterze zdają się łatwiejsze w interpretacji. Różnice ewaluacyjne między danymi zawierającymi różny odsetek błędnie nadanych kodów są stałe niezależnie od liczby danych treningowych i sposobu uczenia się. Warunkiem jest jednak, że tych danych jest wystarczająco dużo, tj. od momentu gdy miara F1 zaczyna wykazywać zbliżoną do liniowej zależność od liczby przypadków treningowych. Zakładając na podstawie ekstrapolacji, że zwiększenie liczby danych nie zmieni zaobserwowanych trendów, można wysnuć wniosek, że więcej błędnie zakodowanych dokumentów prowadzi do pogorszenia działania modelu, rozumianego jako zwiększenie prawdopodobieństwa, że model będzie nadawał nowym danym nieprawidłowe kody.

W przypadku klasyfikatora tematów wyniki analizy regresji liniowej zdają się częściowo potwierdzać wcześniejsze wnioski. Więcej nieprawidłowo zakodowanych przypadków prowadzi do gorszych wyników ewaluacyjnych modelu. Dwie rzeczy wymagają dodatkowego komentarza. Po pierwsze, różnice w wynikach ewaluacji modeli uczonych na różnej jakości danych nie są stopniowe. Różnice F1 między modelami trenowanymi na danych zawierających 100% i 90% poprawnych przypadków są wyraźnie większe, niż gdy dane składają się z 60% i 70% poprawnych przypadków, a model wytrenowany na 50% i 60% poprawnych przypadków w ogóle nie wykazał statystycznie istotnych różnic. Po drugie, z nieznanych przyczyn (wykluczono błąd w danych i błąd w kodzie, wszystkie dane wygenerowano jeszcze raz, aby wyeliminować czynnik losowy) przy obu metodach uczenia się zauważono, że w pewnym momencie wyniki ewaluacji dla modeli zawierających największą liczbę błędnych danych zaczynają poprawiać się znacznie szybciej niż w przypadku lepszych jakościowo danych. Ograniczona liczebność danych uniemożliwia dalszą eksplorację tego trendu, tj. nie wiadomo, jak model zachowywałby się, gdyby dostarczono mu więcej przypadków.

Wykres 1. Miara F1 w zależności od odsetka błędnie zakodowanych przypadków (test na poprawnie zakodowanych przypadkach)



* *W warunku uczenia aktywnego na dużych próbach z niskim odsetkiem poprawnych danych model przestawał się uczyć ze względu na brak dostępnych „trudnych" przypadków. Z tego względu część linii kończy się, zanim osiągnięta jest maksymalna możliwa wielkość zbioru treningowego.*

Tabela 1. Wyniki modelu regresji liniowej 1–2 (Zmienna zależna: miara F1 dla modelu testowanego na poprawnie zakodowanych przypadkach)

Predyktor	Miara F1 dla klasyfikatora stanowiska	Miara F1 dla klasyfikatora tematu
	B	
Odsetek poprawnie zakodowanych przypadków (ref. 100%)		
90%	-0,026***	-0,013***
80%	-0,049***	-0,025***
70%	-0,067***	-0,038***
60%	-0,084***	-0,041***
50%	-0,1***	-0,041***
Liczba przypadków	2,558***	-3,929***
Liczba przypadków^2	0,289***	7,786***
Liczba przypadków^3		-2,754***
Metoda uczenia (ref. uczenie aktywne) uczenie pasywne	0,001	-0,006***
(stała)	0,559***	0,301***
Skorygowane R-kwadrat	0,3	0,64

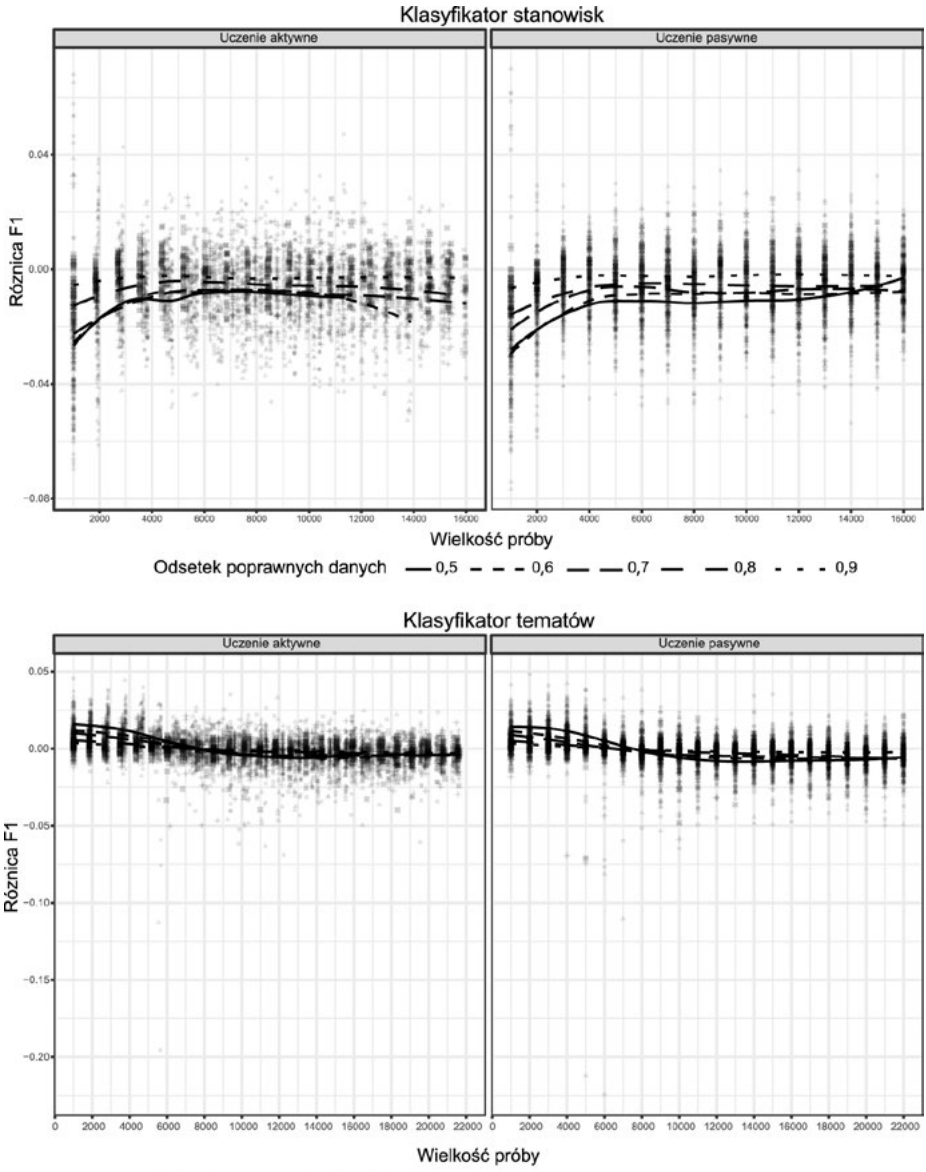
*** $p < 0{,}001$

Wniosek 2. Model uczy się błędnych klasyfikacji

Osobnym zagadnieniem przy dostarczeniu niepoprawnie zakodowanych danych jest skrzywienie algorytmiczne. Może się wydarzyć, że koder lub część koderów źle zrozumie instrukcje kodowania i będzie systematycznie popełniać błędy w interpretacji tekstu i nadawaniu odpowiedniej etykiety. Algorytm sam z siebie nie zna poprawnej interpretacji, lecz uczy się na dostarczonych mu przykładach. Jeśli będą one źle zakodowane, lecz błędy koderów będą systematyczne, to wygeneruje on funkcję, która odpowiada interpretacji koderów. Z punktu widzenia badacza istotne jest, czy takie błędy interpretacyjne da się zaobserwować, np. porównując modele wytrenowane na danych pochodzących od różnych koderów albo zawierające różny odsetek niepoprawnie zakodowanych przypadków. Zgodnie z regułami sztuki trenowania modeli uczenia maszynowego dane te można losowo podzielić na treningowe i testowe, i na tej podstawie dokonać ewaluacji. Błędy byłyby możliwe do wykrycia, gdyby każdy taki zbiór dawał różne wyniki w ewaluacji.

Analiza regresji liniowej sugeruje, że różnice miary F1 dla modeli wytrenowanych na danych o różnym stopniu poprawności są istotne statystycznie,

Wykres 2. Różnica w mierze F1 między modelami wytrenowanymi na poprawnie i błędnie zakodowanych danych



ale największa wynosi 0,012 punktu (dla klasyfikatora stanowisk, dla klasyfikatora tematów maksymalna różnica wyniosła 0,002). Interpretacja danych na wykresie 2 skłania do wniosku, że przy tak niewielkich różnicach średnich

i obserwowanym odchyleniu standardowym właściwie niemożliwe jest wyciąganie konkluzji na temat jakości danych treningowych. Algorytm fastText osiąga w zasadzie nierozróżnialne wyniki ewaluacji. Oznacza to, że uczy się popełniać te same błędy interpretacyjne co koderzy i w realnych badaniach może sobie doskonale radzić w nieprawidłowej (stosując za kryterium intencje autora książki kodowej) automatycznej klasyfikacji.

Tabela 2. Wyniki modelu regresji liniowej 3-4 (Zmienna zależna: miara F1 dla modelu testowanego na niepoprawnie zakodowanych przypadkach)

Predyktor	Różnica F1 dla stanowiska	Różnica F1 dla tematu
	B	
Odsetek poprawnie zakodowanych przypadków (ref. 100%)		
90%	-0,003***	-0,001**
80%	-0,006***	-0,001***
70%	-0,009***	-0,002***
60%	-0,011***	-0,001***
50%	-0,012***	0
Liczba przypadków	0,149***	-0,409***
Liczba przypadków^2	-0,19***	0,218***
Metoda uczenia (ref. uczenie aktywne) uczenie pasywne	0	-0,001***
(stała)	0	0,001**
Skorygowane R-kwadrat	0,16	0,14

*** $p < 0{,}001$

Wniosek 3. Liczba przypadków ma większe znaczenie niż jakość (pod pewnymi warunkami)

Dotychczasowe analizy dostarczyły dowodów, że zwiększenie odsetka niepoprawnie zakodowanych tweetów prowadzi do pogorszenia działania sieci neuronowych. Natomiast nie zostało ustalone, czy lepiej jest w związku z tym sprawdzać poprawność kodowania, poprzez dwukrotne zakodowanie każdego przypadku, czy też dwukrotne zwiększenie liczby danych treningowych poprzez jednokrotne kodowanie przypadków kosztem tego, że niektóre z nich z pewnością będą zakodowane niepoprawnie.

Analiza regresji wskazuje, że im mniej błędów koderzy popełniają, tym korzystniejsza jest decyzja polegająca na jednokrotnym kodowaniu danych. Jest to zgodne z argumentacją Barbery i in. (2021). Na wykresie 3 zauważyć można, jaki

dokładnie jest próg tolerancji, aby taka rekomendacja miała uzasadnienie. Ponownie wyniki są odmienne dla klasyfikatora stanowisk i klasyfikatora tematów.

W przypadków pierwszego zadania jedyny zysk w działaniu modelu następuje, gdy koderzy są bezbłędni. Wówczas faktycznie nie ma uzasadnienia, aby kodować ten sam dokument więcej niż raz. W przypadku, gdy dane treningowe zawierają 10% błędów, model o ustabilizowanej krzywej uczenia się uczy się tak samo na danych zweryfikowanych jak i dwukrotnie większych danych niezweryfikowanych. W przypadku, gdy koderzy się mylą częściej niż w 10%, rekomendowana jest weryfikacja kodów kosztem liczebności danych treningowych.

W przypadku klasyfikatora tematów interpretacja wyniku wymaga odwołania się do wykresu 1. Model początkowo dobrze sobie radzi w klasyfikowaniu tweetów, po czym następuje gwałtowny spadek miary F1. Mogło to być spowodowane tym, że dopiero przy kilku tysiącach przypadków, wszystkie, tj. także te najmniej liczne klasy ukazały się w pełni swojej złożoności. W konsekwencji dopiero przy ok. 11000 przypadków model zaczął uczyć się coraz lepiej rozpoznawać także te najrzadziej występuje klasy i od tego momentu faktycznie zaobserwować można, że miara F1 rośnie wraz z liczbą przypadków (analogicznie jest to sytuacja obserwowana przy klasyfikacji stanowisk na próbie do 2000–4000 przypadków). Jest to jednak także punkt oznaczający limit dostępnych danych. Koderzy w zakresie tematów popełnili 11225 błędów, co oznacza, że możliwe jest zasymulowanie wyników jedynie dla nieco ponad 22000 przypadków zakładając, że połowa z nich jest nieprawidłowa. Problem interpretacyjny polega na tym, że przy tak trudnym zadaniu na tym początkowym etapie liczba przypadków wcale nie prowadzi do polepszenia statystyk ewaluacyjnych modelu (a przynajmniej nie miary F1). Porównanie miary F1 przy 5000 i 10000 przypadków oraz 11000 i 22000 przypadków jest nieuprawnione, bo model jest nie tyle na innym etapie uczenia się, ile właściwie w zupełnie innej jego fazie (rozpoznania danych i klas, a nie poprawiania klasyfikacji). Do wyciągnięcia wniosków należałoby użyć właściwie wyników modeli uczących się na więcej niż 11000 zweryfikowanych przypadkach. Wówczas mogłyby one być zbliżone do tego, co zaobserwowano w modelu klasyfikującym stanowiska, tj. że jednokrotne kodowanie ma uzasadnienie tylko i wyłącznie, gdy koderzy mylą się rzadziej niż w 10% przypadków. Są dwie przesłanki, które sprawiają, że taka konkluzja ma uzasadnienie. Po pierwsze, wyniki analizy regresji w tabeli 3 wskazują, że im więcej błędów popełniają koderzy, tym bardziej ryzykowne jest pozostawienie ich pracy bez weryfikacji. Po drugie, wyniki uzyskane na maksymalnej możliwej liczbie danych treningowych dla każdego poziomu błędnych kodów wskazują, że wraz z rosnącą liczbą przypadków linia trendu dąży do przecięcia punktu 0 na osi Y (por. wykres A1 w Aneksie). Inaczej mówiąc, im więcej danych klasyfikator otrzymuje, tym bardziej uzasadnione jest weryfikowanie danych.

Wykres 3. Miara F1, gdy każdy przypadek był zakodowany przez dwóch koderów w porównaniu do miary F1 na dwukrotnie większych danych zakodowanych jednokrotnie (w obu przypadkach zbiór testowy był zweryfikowany pod kątem poprawności)

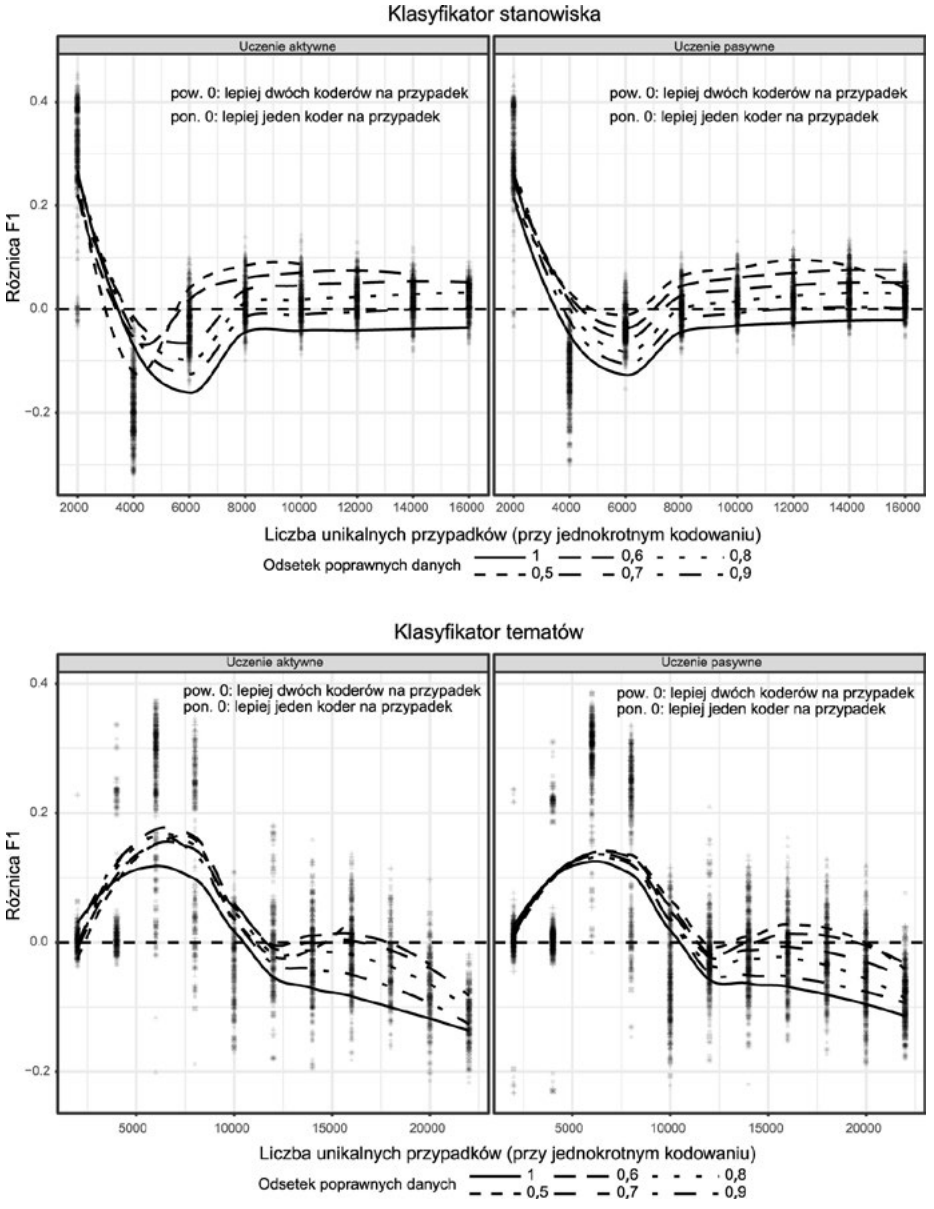


Tabela 3. Wyniki modelu regresji liniowej 5–6 (Zmienna zależna: Różnica między miarą F1 obliczoną na zbiorze dwukrotnie zakodowanych przypadków a miarą F1 obliczoną na dwukrotnie większym zbiorze jednokrotnie zakodowanych przypadków)

Predyktor	Różnica F1 dla stanowiska	Różnica F1 dla tematu
	B	
Odsetek poprawnie zakodowanych przypadków (ref. 100%)		
90%	0,028***	0,017***
80%	0,052***	0,029***
70%	0,069***	0,043***
60%	0,082***	0,051***
50%	0,096***	0,048***
poly(Liczba przypadków, 3)1	-1,448***	-4,645***
poly(Liczba przypadków, 3)2	3,588***	-1,623***
poly(Liczba przypadków, 3)2	-4,738***	3,013***
Metoda uczenia (ref. uczenie aktywne) uczenie pasywne	0,007**	0,002
(stała)	-0,026***	-0,005
Skorygowane R-kwadrat	0,54	0,36

*** $p < 0{,}001$

Konkluzje i rekomendacje

Przeprowadzone analizy pozwoliły określić, w jakim stopniu brak weryfikacji pracy koderów wpływa na działanie klasyfikatorów tekstu. Im więcej błędnie zakodowanych przypadków zawierają dane treningowe, tym większe prawdopodobieństwo, że model będzie dokonywał błędnych klasyfikacji. Taki wynik jest raczej potwierdzeniem przypuszczeń opartych na logicznych przesłankach niż czymś zaskakującym. Warto mieć jednak na uwadze, z czym się wiąże w realnym procesie badawczym. Badacz, który zdecyduje zgodnie z niektórymi rekomendacjami (Barberá i in. 2021), że ważniejsze jest zwiększenie wielkości zbioru treningowego kosztem weryfikacji jakości kodowania, zazwyczaj nie ma możliwości cofnięcia swojej decyzji (jeśli nie dysponuje rezerwami finansowymi lub możliwościami zdobycia dodatkowych środków, które pozwolą na kontrolę jakości danych). Może oczywiście wypróbowywać inne modele nadzorowane, jak też sprawdzać, jakie będą efekty zastosowania różnych procedur wstępnego przetwarzania danych (lematyzacja, usunięcie interpunkcji, emotikonów, emoji

itp.). Nie może jednakże zweryfikować, czy osiągane wyniki działania modelu są optymalne, czy też poniżej optimum ze względu na niską jakość danych treningowych. Symulacje z kolei pokazują, że nawet 10% błędnie zakodowanych danych może prowadzić do zauważalnego spadku miary F1.

Wyniki można odczytywać jako niepokojące ze względu na część analiz dotyczących wpływu błędnie zakodowanych przypadków na działanie klasyfikatora. Okazuje się, że niezależnie od tego, jaki odsetek danych treningowych i testowych jest błędny, wyniki ewaluacji są od siebie nieodróżnialne. Inaczej mówiąc, badacz nie jest w stanie się zorientować, który koder popełnia więcej błędów, ponieważ algorytm uczy się klasyfikować zgodnie z nieprawidłowymi interpretacjami. W konsekwencji może to doprowadzić do powstania pozornie dobrze działającego klasyfikatora, który będzie porządkował tekst osiągając wysokie statystyki ewaluacyjne, ale niezgodnie z założeniami, które określił badacz. Bez weryfikacji jakości danych wykrycie takich nieprawidłowości jest niemożliwe. Biorąc zaś pod uwagę to, że zazwyczaj klasyfikator nie jest w badaniach społecznych celem samym w sobie, lecz narzędziem do osiągnięcia innych celów badawczych, to rezultatem mogą być wątpliwej jakości wyniki badań empirycznych.

Rekomendacja, aby na jeden kodowany przypadek przypadał tylko jeden koder, ma uzasadnienie tylko wtedy, gdy jest absolutna pewność co do jakości pracy koderów. Oznacza to, że koderzy muszą być doświadczeni, przeszkoleni w interpretacji konkretnej książki kodowej i godni zaufania, tj. zachowywać najwyższe standardy pracy przy świadomości, że nie będzie ona kontrolowana. W rzeczywistości sytuacja, w której powstała w ten sposób baza danych jest bezbłędna, pozostaje w kategoriach celu, do którego należy dążyć czy też założenia, nie zaś realnego warunku (Hand 2006). Na przeszkodzie stoją między innymi ograniczenia ludzkiego organizmu jak też złożoność danych. Nawet najbardziej doświadczony i sumienny koder się męczy, rozprasza i popełnia nieintencjonalne błędy. Ponadto, przekaz sformułowany w języku naturalnym może być niejednoznaczny i mimo dużego doświadczenia koderzy mogą napotkać na trudności w interpretacji tego, czego dotyczy wypowiedź. Nawet tak relatywnie prosta kwestia jak zajęcie stanowiska w określonej sprawie, może być wyrażone wprost, jak też w sposób niejasny, ironiczny, przy zastosowaniu nieprawidłowo użytych słów, co zmusza kodera nieraz do daleko idących interpretacji (Joseph i in. 2021). Z kolei im bardziej złożone kwestie są poruszane, tym większe prawdopodobieństwo, że koderzy będą je odmiennie interpretować (DiMaggio 2015). Podsumowując tę część wniosków, można powiedzieć, że ryzyko związane z jednokrotnym kodowaniem jest uzasadnione tylko w sytuacjach, gdy spodziewać się można niewielkiej liczby błędów (według analiz poniżej 10%). Są to zatem z konieczności zadania niewymagające interpretacji i proste, tj. dotyczące niewielkiej liczby klas.

Rekomendacje wbrew pozorom wcale nie są łatwe do ustalenia. Więcej danych to zazwyczaj lepiej działający model. Tylko, czy ma to znaczenie, jeśli model klasyfikuje niezgodnie z intencją badaczy? Mniej danych to zazwyczaj gorsze wyniki działania, a więc i w tym wypadku model także będzie klasyfikować dokumenty niezgodnie z pierwotnymi celami. Odpowiedzią mogą być rozwiązania pośrednie, które pozwolą z jednej strony w ramach tych samych środków pozyskać więcej danych treningowych niż w przypadku, gdy na jeden dokument przypada co najmniej dwóch koderów, ale też zachować część kontroli nad jakością bazy danych. Wśród nich wymienić można:

- Losowe zdublowanie części przypadków (np. 15–20%), które koduje ten sam koder. Celem jest sprawdzenie, czy ta sama osoba koduje dane w spójny sposób, tj. weryfikacja umiejętności kodera i identyfikacja kodów sprawiających trudności.
- Losowe przypisanie określonego odsetka tych samych przypadków (np. 15–20%) do różnych koderów, aby sprawdzić, jaki jest poziom zgodności między koderami. W ten sposób również można zidentyfikować błędy kodowania, które można skorygować poprzez dodatkowe szkolenia. Ponadto możliwe jest także ustalenie skali błędów. Przykładowo jeśli w wylosowanym kontrolowanym zbiorze danych różnice obejmują kilkadziesiąt procent, to jest to sygnał, że algorytm, który może mimo wszystko osiągać wysokie wskaźniki ewaluacyjne, klasyfikuje niezgodnie z intencjami badacza.
- Weryfikowanie poprawności kodowania w bazie testowej. Wyniki wskazały, że statystyki ewaluacyjne modelu są bardzo podobne niezależnie od tego, jak duży jest odsetek niepoprawnych danych treningowych. W związku z tym, ewentualne nieprawidłowości w uczeniu się modelu mogą zostać dostrzeżone jedynie, jeśli wyniki modelu są sprawdzane na zweryfikowanej bazie testowej zakodowanej zgodnie z intencjami badaczy (por. tabela 2).

Analizy przeprowadzone w tym artykule posiadają ograniczenia, o których należy wspomnieć. Po pierwsze, zastosowany został tylko algorytm fastText. Choć wydaje się, że wnioski powinny być podobne przy zastosowaniu innych sieci neuronowych, to jednak skala różnic ewaluacyjnych w zależności od odsetka nieprawidłowych kodów może być inna. Po drugie, kwestia tego, jaki jest akceptowalny poziom nieprawidłowych kodów przy skomplikowanych klasyfikacjach nie została jednoznacznie empirycznie ustalona, a jedynie wywnioskowana na podstawie wyników analizy regresji liniowej i wizualizacji dostępnych danych. Weryfikacja empiryczna wymagałaby prawdopodobnie przynajmniej dwukrotnie większej liczby danych treningowych.

Bibliografia

Ajdukiewicz, Kazimierz. 1965. *Logika pragmatyczna*. Warszawa: Państwowe Wydawnictwo Naukowe.

Anon. 2021. „FastText". *Facebook Research*. Pobrano 17 marzec 2021 (https://research.fb.com/downloads/fasttext/).

Bai, Qiyu, Qi Dan, Zhe Mu, Maokun Yang. 2019. A Systematic Review of Emoji: Current Research and Future Perspectives. *Frontiers in Psychology*. DOI: 10.3389/fpsyg.2019.02221.

Bail, Christopher A. 2014. The Cultural Environment: Measuring Culture with Big Data. *Theory and Society*, 43, 3: 465–82. DOI: 10.1007/s11186-014-9216-5.

Bakliwal, Akshat, Jennifer Foster, Jennifer van der Puil, Ron O'Brien, Lamia Tounsi, Mark Hughes. 2013. Sentiment Analysis of Political Tweets: Towards an Accurate Classifier. In: *Proceedings of the NAACL Workshop on Language Analysis in Social Media*. Atlanta, GA.: Association for Computational Linguistics.

Barberá, Pablo, Amber E. Boydstun, Suzanna Linn, Ryan McMahon, Jonathan Nagler. 2021. Automated Text Classification of News Articles: A Practical Guide. *Political Analysis*, 29, 1:19–42. DOI: 10.1017/pan.2020.8.

Batorski, Dominik, Marta Olcoń-Kubicka. 2006. Prowadzenie badań przez internet – podstawowe zagadnienia metodologiczne. *Studia Socjologiczne*, 182, 3: 99–132.

Batorski, Dominik, Krzysztof Olechnicki. 2007. Wprowadzenie do socjologii internetu. *Studia Socjologiczne*, 186, 3: 5–14.

Boudon, Raymond. 1997. *The Art of Self-Persuasion: The Social Explanation of False Beliefs*. Cambridge, England; Malden, Mass.: Polity.

Brants, Wesley, Bonita Sharif, Alexander Serebrenik. 2019. Assessing the Meaning of Emojis for Emotional Awareness – A Pilot Study. s. 419–23. In: *Companion Proceedings of The 2019 World Wide Web Conference*, *WWW '19*. New York, NY, USA: Association for Computing Machinery.

Cha, Meeyoung, Hamed Haddadi, Fabrício Benevenuto, Krishna P. Gummadi. 2010. Measuring user influence in Twitter: The million follower fallacy. In: *ICWSM '10: Proceedings of international AAAI Conference on Weblogs and Social*.

Chen, Yukun, Subramani Mani, Hua Xu. 2012. Applying Active Learning to Assertion Classification of Concepts in Clinical Text. *Journal of Biomedical Informatics*, 45, 2: 265–72. DOI: 10.1016/j.jbi.2011.11.003.

Denny, Matthew J., Arthur Spirling. 2018. Text Preprocessing For Unsupervised Learning: Why It Matters, When It Misleads, And What To Do About It. *Political Analysis*, 26, 2: 168–89. DOI: 10.1017/pan.2017.44.

Di Franco, Giovanni, Michele Santurro. 2020. Machine Learning, Artificial Neural Networks and Social Research. *Quality & Quantity*. DOI: 10.1007/s11135-020-01037-y.

DiMaggio, Paul. 2015. Adapting Computational Text Analysis to Social Science (and Vice Versa). *Big Data & Society*, 2, 2. DOI: 10.1177/2053951715602908.

Drus, Zulfadzli, Haliyana Khalid. 2019. Sentiment Analysis in Social Media and Its Application: Systematic Literature Review. *Procedia Computer Science*, 161: 707–14. DOI: 10.1016/j.procs.2019.11.174.

Goldenstein, Jan, Philipp Poschmann. 2019. A Quest for Transparent and Reproducible Text-Mining Methodologies in Computational Social Science. *Sociological Methodology*, 49, 1: 144–51. DOI: 10.1177/0081175019867855.

Grimmer, Justin, Brandon M. Stewart. 2013. Text as Data: The Promise and Pitfalls of Automatic Content Analysis Methods for Political Texts. *Political Analysis*, 21, 3: 267–97. DOI: 10.1093/pan/mps028.

HaCohen-Kerner, Yaakov, Daniel Miller, Yair Yigal. 2020. The Influence of Preprocessing on Text Classification Using a Bag-of-Words Representation. *PLOS ONE*, 15, 5: e0232525. DOI: 10.1371/journal.pone.0232525.

Haddi, Emma, Xiaohui Liu, Yong Shi. 2013. The Role of Text Pre-Processing in Sentiment Analysis. *Procedia Computer Science*, 17: 26–32. DOI: 10.1016/j.procs.2013.05.005.

Hand, David J. 2006. Classifier Technology and the Illusion of Progress. *Statistical Science*, 21, 1: 1–14. DOI: 10.1214/088342306000000060.

He, Zhoushanyue, Matthias Schonlau. 2020a. Automatic Coding of Open-Ended Questions into Multiple Classes: Whether and How to Use Double Coded Data. *Survey Research Methods*, 14, 3: 267–87. DOI: 10.18148/srm/2020.v14i3.7639.

He, Zhoushanyue, Matthias Schonlau. 2020b. Automatic Coding of Text Answers to Open-Ended Questions: Should You Double Code the Training Data? *Social Science Computer Review*, 38, 6: 754–65. DOI: 10.1177/0894439319846622.

Hopkins, Daniel J., Gary King. 2010. A Method of Automated Nonparametric Content Analysis for Social Science. *American Journal of Political Science*, 54, 1: 229–47. DOI: 10.1111/j.1540-5907.2009.00428.x.

Ignatow, Gabe. 2016. Theoretical Foundations for Digital Text Analysis. *Journal for the Theory of Social Behaviour*, 46, 1: 104–20. DOI: 10.1111/jtsb.12086.

Jemielniak, Dariusz. 2018. Socjologia 2.0: o potrzebie łączenia Big Data z etnografią cyfrową, wyzwaniach jakościowej socjologii cyfrowej i systematyzacji pojęć. *Studia Socjologiczne*, 242, 2: 7–29. DOI: 10.24425/122461.

Jemielniak, Dariusz. 2019. *Socjologia internetu*. Warszawa: Wydawnictwo Naukowe Scholar.

Jordan, Michael, Tom Mitchell. 2015. Machine Learning: Trends, Perspectives, and Prospects. *Science*, 349, 6245: 255–60. DOI: 10.1126/science.aaa8415.

Joseph, Kenneth, Sarah Shugars, Ryan Gallagher, Jon Green, Alexi Quintana Mathé, Zijian An, David Lazer. 2021. (Mis)alignment Between Stance Expressed in Social Media Data and Public Opinion Surveys. *arXiv:2109.01762 [cs]*.

Joulin, Armand, Edouard Grave, Piotr Bojanowski, Tomas Mikolov. 2016. Bag of Tricks for Efficient Text Classification. *arXiv:1607.01759 [cs]*.

Krippendorff, Klaus H. 2003. *Content Analysis: An Introduction to Its Methodology*. Thousand Oaks, Calif: Sage Publications, Inc.

Lazer, David, Alex (Sandy) Pentland, Lada Adamic, Sinan Aral, Albert Laszlo Barabasi, Devon Brewer, Nicholas Christakis, Noshir Contractor, James Fowler, Myron

Gutmann, Tony Jebara, Gary King, Michael Macy, Deb Roy, Marshall Van Alstyne. 2009. Life in the network: the coming age of computational social science. *Science*, 323, 5915: 721–23. DOI: 10.1126/science.1167742.

Lin, Chenghua, Yulan He. 2009. Joint sentiment/topic model for sentiment analysis. In: *Proceedings of the 18th ACM conference on Information and knowledge management*, *CIKM '09*. New York, NY, USA: Association for Computing Machinery, 375–384.

Marciszewski, Witold. 1972. *Podstawy logicznej teorii przekonań*. Warszawa: Państwowe Wydawnictwo Naukowe.

Miller, Blake, Fridolin Linder, Walter R. Mebane. 2020. Active Learning Approaches for Labeling Text: Review and Assessment of the Performance of Active Learning Approaches. *Political Analysis*, 28, 4: 532–51. DOI: 10.1017/pan.2020.4.

Mohammad, Saif M., Parinaz Sobhani, Svetlana Kiritchenko. 2016. Stance and Sentiment in Tweets. *arXiv:1605.01655 [cs]*.

Monroe, Burt L. 2019. The Meanings of "Meaning in Social Scientific Text Analysis. *Sociological Methodology*, 49, 1: 132–39. DOI: 10.1177/0081175019865231.

Mozetič, Igor, Miha Grčar, Jasmina Smailović. 2016. Multilingual Twitter Sentiment Classification: The Role of Human Annotators. *PLOS ONE* 11, 5:e0155036. DOI: 10.1371/journal.pone.0155036.

Murthy, Dhiraj, Sawyer A. Bowman. 2014. Big Data Solutions on a Small Scale: Evaluating Accessible High-Performance Computing for Social Research: *Big Data & Society*. DOI: 10.1177/2053951714559105.

Nelson, Laura K. 2019. To Measure Meaning in Big Data, Don't Give Me a Map, Give Me Transparency and Reproducibility. *Sociological Methodology*, 49, 1: 139–43. DOI: 10.1177/0081175019863783.

Rodak, Olga. 2017. Twitter jako przedmiot badań socjologicznych i źródło danych społecznych: perspektywa konstruktywistyczna. *Studia Socjologiczne*, 226, 3: 209–36.

Salganik, Matthew J. 2017. *Bit by Bit: Social Research in the Digital Age*. Illustrated edition. Princeton: Princeton University Press.

Sobhani, Parinaz, Diana Inkpen, Xiaodan Zhu. 2019. Exploring Deep Neural Networks for Multitarget Stance Detection. *Computational* Intelligence, 35, 1: 82–97. DOI: 10.1111/coin.12189.

Subedi, Nishan. 2018. FastText: Under the Hood. *Medium*. Pobrano 3 grudzień 2021 (https://towardsdatascience.com/fasttext-under-the-hood-11efc57b2b3).

Tharwat, Alaa. 2020. Classification assessment methods. *Applied Computing and Informatics* ahead-of-print(ahead-of-print). DOI: 10.1016/j.aci.2018.08.003.

Tomanek, Krzysztof. 2017. Metodyka dla analizy treści w projektach stosujących techniki text mining i rozwiązania CAQDAS piątej generacji. *Przegląd Socjologii Jakościowej*, 13, 2: 128–43.

Turner, Anna, Marcin W. Zieliński, Kazimierz M. Słomczyński. 2018. Google Big Data: charakterystyka i zastosowanie w naukach społecznych. *Studia Socjologiczne*, 231, 4: 49–71. DOI: 10.24425/122482.

Watts, Duncan J., Peter Sheridan Dodds. 2007. Influentials, Networks, and Public Opinion Formation. *Journal of Consumer Research*, 34, 4: 441–58. DOI: 10.1086/518527.

Wiedemann, Gregor. 2019. Proportional Classification Revisited: Automatic Content Analysis of Political Manifestos Using Active Learning. *Social Science Computer Review*, 37, 2: 135–59. DOI: 10.1177/0894439318758389.

Ziółkowski, Marek. 1989. *Wiedza, jednostka, społeczeństwo: zarys koncepcji socjologii wiedzy*. Warszawa: Państwowe Wydawnictwo Naukowe.

Żulicki, Remigiusz. 2017. Potencjał Big Data w badaniach społecznych. *Studia Socjologiczne*, 226, 3: 175–207.

Aneks

Wykres A1. Różnica między miarą F1 na zweryfikowanym i niezweryfikowanych zbiorze danych treningowych. Uczenie pasywne