Zachodniopomorski Uniwersytet Technologiczny w Szczecinie

Wydział Informatyki

mgr inż. Marcin Pietras

Syntaktyczna i semantyczna analiza danych tekstowych z wykorzystaniem modeli Markowa realizowanych sprzętowo

Streszczenie pracy doktorskiej

Promotor: dr hab. inż. Przemysław Klęsk, prof. ZUT

Szczecin 2018

Spis treści

Finansowanie prac badawczych

Prezentowane badania nad syntaktyczną i semantyczną analizą danych tekstowych z wykorzystaniem modeli Markowa realizowanych sprzętowo wykonane zostały przy wsparciu z dotacji celowej numer: 517–05–023–4781/17, przyznanej przez Ministerstwo Nauki i Szkolnictwa Wyższego.

1 Aktualność problemu

Analiza syntaktyczna i semantyczna odgrywają ważną rolę w automatyzacji przetwarzania danych tekstowych. Przeważnie rozpoznawanie kategorii syntaktycznych dokonywane jest w pierwszej kolejności, a na podstawie otrzymanej struktury składniowej wykonywana jest analiza semantyczna. Podobnie jest również w rozprawie, gdzie na podstawie otrzymanej struktury składniowej dokonywana jest analiza pola semantycznego wybranego wyrazu (ang. *Word of Interest*) w badanym tekście. Na przełomie ostatnich dwóch dekad naukowcy zaproponowali wiele systemów dokonujących syntaktyczno-semantycznej analizy o różnorodnej specjalizacji tematycznej (Bos 2011, Cambria & White 2014). Wciąż poszukiwane są coraz nowsze metody wykorzystujące techniki uczenia maszynowego zarówno z nadzorem, jak i bez. Największy postęp w tej dziedzinie współgra z popularnością języka, dla którego dany system jest rozwijany – nie dziwi więc fakt, że dla języka angielskiego dostępne są najbogatsze bazy danych (tj. PropBank (Palmer et al. 2006), NomBank (Meyers Adam & Macleod 2008), Penn Treebank (Taylor et al. 2003)) odpowiednio oznakowanych tekstów (syntaktycznie, ale i semantycznie). Dla mniej popularnych języków tworzone są również podobne bazy danych (np. Składnica) głównie za sprawą rozwoju technik parsowania i projekcji oznaczeń z innego języka (Pado & Lapata 2009). Nadal jednak oznaczanie danych tekstowych częściami mowy i częściami zdania jest skomplikowanym zadaniem wymagającym intensywnych obliczeń. Choć nowoczesne techniki często pozwalają osiągnąć wystarczającą dokładność analizy, to wciąż jest wiele do zrobienia w kierunku optymalizacji stosowanych metod.

Wielu badaczy (Lluis & Marquez 2008, Lluis 2008, Dai et al. 2009, Samuelsson et al. 2008, Llus et al. 2013) uznało, że system realizujący interakcje między analizą składniową i semantyczną umożliwia uzyskanie jakościowo lepszego wyniku niż w analogicznych systemach, w których te czynności są wykonywane oddzielnie.

Dziś jest oczywiste, że z powodu niejasności w konstrukcji języka naturalnego (Przepiórkowski et al. 2012), budowa drzew składniowych z dużą dokładnością nie może ograniczać się tylko do znajomości gramatyki języka dotyczącej części mowy i cech morfologicznych. Niejednoznaczność języka naturalnego wymaga opracowania specjalnych analizatorów wykorzystujących metody statystyczne i podejścia z użyciem technik uczenia maszynowego. W szczególności, Przepiórkowski et al. (2012, str. 52) stwierdzają, że

> *„(...) W wypadku korpusów znacznie większych, takich jak należący obecnie do największych na świecie Narodowy Korpus Języka Polskiego, jedyną możliwością jest anotacja automatyczna, za pomocą specjalnie do tego stworzonych programów komputerowych. Programy takie muszą się jednak nauczyć znakować teksty, a do tego potrzebują. . . tekstów znakowanych przez człowieka. Stąd właśnie potrzeba stworzenia ręcznie anotowanego podkorpusu NKJP (...)“*

oraz

> *„(...) Bezpośrednie sformalizowanie lingwistycznych reguł znakowania okazuje się zadaniem znacznie trudniejszym niż stworzenie takiego korpusu treningowego (...)“.*

Dlatego też w dzisiejszych parserach tekstów w języku naturalnym często analizowana jest nie tylko informacja o części mowy, cechach morfologicznych i cechach strukturalnych, ale także znaczenie leksykalne wyrazów, co w pewnym sensie jest formą znajomości semantyki języka (Baroni & Zamparelli 2010).

Do dalszego rozwoju metod analizy syntaktycznej i semantycznej przyczynili się uczestnicy konkursu CoNLL Shared Task 2008 (Surdeanu et al. 2008) oraz CoNLL Shared Task 2009 (Hajic et al. 2009). W 2008 roku analiza dotyczyła tylko języka angielskiego. W 2009 roku uczestnicy dokonali analizy sześciu kolejnych języków: katalońskiego, chińskiego, czeskiego, japońskiego, niemieckiego i hiszpańskiego. Sześć drużyn biorących udział w konkursie w 2008 roku i cztery zespoły w 2009 roku zastosowały metody, które w pewien sposób łączyły metody analizy składniowej i semantycznej.

W pracach (Lluis & Marquez 2008, Lluis 2008) opisano system, który został zgłoszony do konkursu w 2008 roku, a w pracy (Lluis et al. 2009) przedstawiono modyfikacje systemu, który brał udział w konkursie rok później. Syntaktyczna i semantyczna analiza w tych systemach była realizowana poprzez budowanie drzewa zależności równocześnie z utworzeniem etykiet relacji semantycznych. Do wstępnego i głównego parsera zastosowano algorytm Eisnera (Eisner 1996). Badania eksperymentalne nad tą metodą wykazały, że połączona analiza nie wpływa na wyniki parsera syntaktycznego, ale poprawia jakość rozpoznawania semantycznego w stosunku do systemu, w którym analizy syntaktyczna i semantyczna są wykonywane osobno.

System opisany w pracy (Henderson et al. 2008) przedstawia analizator korzystający z metody „shift-reduce“[1] z równoległą budową konstrukcji składniowej i semantycznej zdania. W celu predykcji struktury zależności dla analizatora przeszkolony został model oparty o dynamiczną sieć Bayesa – zmieniającą swoją strukturę na podstawie wstępnie określonego stanu wyjściowego (Titov & Henderson 2007). Doświadczenia przeprowadzone w badaniach wykazały, że w przypadku braku współpracy pomiędzy analizą syntaktyczną i semantyczną znacznemu pogarszaniu ulega jakość analizy semantycznej. W pracach (Gesmundo et al. 2009, Henderson et al. 2013) autorzy wykazali również, że integracja analizy syntaktycznej i semantycznej może nieznacznie obniżać jakość tej pierwszej. W pracy (Samuelsson et al. 2008) zastosowano połączenie ośmiu parserów skonstruowanych na podstawie rożnych konfiguracji systemu MaltParsera, co pozwoliło na znaczne zwiększenie jakości analizy w porównaniu z wynikami najlepszego pojedynczego analizatora. W pracy (Dai et al. 2009) wdrożone zostało podejście, w którym rozpoznawanie kategorii syntaktycznych i semantycznych wykonywane jest sekwencyjnie. Eksperymenty wykazały, że takie podejście może również znacznie poprawić efekt analizy. Autorzy sformułowali wniosek, że efekt dalszej analizy zależy od jakości wstępnego rozpoznawania kategorii syntaktycznych. Z kolei Sun et al. (2008) realizowali podejście, które łączy w sobie etapy wstępnej analizy składniowej i semantyczne poszukiwania argumentów za pomocą modelu Markowa z wykorzystaniem maksymalnej entropii (McCallum et al. 2000).

Wśród ostatnich publikacji dotyczących analizy składniowej i semantycznej, oprócz prac ukazujących rezultaty seminarium CoNLL Shared Task należy zauważyć pozycję (Llus et al. 2013), która proponuje podejście do maksymalizacji wspólnej oceny struktur składniowych i semantycznych. Doprowadziło to do bardziej precyzyjnego określenia funkcji semantycznych w porównaniu z układem, w którym te dwa typy analizy prowadzone są oddzielnie.

Jak widać, stosowane przez badaczy metody wykraczają daleko poza proste techniki zliczania tokenów, kolokacji i semantycznych klas słów na etapie analizowania. Niektórzy badacze proponują podejście alternatywne do tradycyjnego, kiedy to analiza syntaktyczna budująca drzewo składniowe ściśle poprzedza analizę semantyczną (która stanowi reprezentację znaczenia tekstu) i łączą te dwa rodzaje analiz w ramy jednej procedury. Istnieją parsery od razu budujące głębokie semantyczne struktury, bez struktur składniowych pośrednich – co całkowicie usuwa granice między analizą składniową i semantyczną (Grefenstette et al. 2014).

Badania z zakresu lingwistyki komputerowej koncentrują się nad rozwiązaniami pozwalającymi na skuteczne przeprowadzenie wysoce dokładnej analizy tekstów zapisanych w języku naturalnym. W dynamicznie rozwijającej się dziedzinie uczenia maszynowego poszukuje się innowacyjnego podejścia wykonującego obliczenia w najefektywniejszy sposób. Dobrym przykładem

[1]Parser przesuwa (ang. *shift*) symbole na stos. Jeśli symbole ze stosu pasują do prawidłowej reguły gramatycznej w bieżącym kontekście, wówczas analizator składni redukuje (ang. *reduce*) je zastępując symbolem nieterminalnym.

jest tu dziedzina nauczania sztucznych sieci neuronowych przechodząca swój renesans za sprawą głębokiego uczenia (ang. *deep learning*), które potwierdziło swoją skuteczność w wielu dziedzinach rozpoznawania wzorców (Lecun et al. 2015, Schmidhuber 2015). Jednakże nowe podejście – choć skuteczne – jest wyjątkowo wymagające obliczeniowo (obliczanie odpowiedzi wielowarstwowej sieci liczącej często tysiące neuronów w każdej z warstw, w których przetwarzanie informacji dokonywane jest na różnych poziomach abstrakcji[2]). Dlatego też wiele ośrodków badawczych i przemysłowych prowadzi intensywne badania w celu stworzenia nowych wydajnych rozwiązań sprzętowych (Gupta et al. 2015a, Zhang et al. 2015, Ovtcharov et al. 2015). Wykorzystując dedykowane rozwiązania sprzętowe możliwe jest osiągnięcie wysokiego stopnia zrównoleglenia algorytmów uczenia maszynowego (Pietras 2014). Szczególne znaczenie ma to dla rozwiązań stosujących chmury obliczeniowe (ang. *cloud computing*). Przykładowo firma Google budowę zaplecza swoich usług z zakresu uczenia maszynowego i sztucznej inteligencji oparła na własnym wyspecjalizowanym chipie TPU (ang. *Tensorflow Processing Unit*) (Yunfei 2017), podczas gdy firma Microsoft zdecydowała się na zastosowanie układów FPGA (ang. *Field-Programmable Gate Array* – bezpośrednio programowalna macierz bramek). Dzięki rekonfigurowalności[3] tych układów – czyli możliwości zmiany struktury obliczeniowej w trakcie pracy układu – łatwo przełączyć kontekst aplikacji. Oznacza to, że w jednej chwili możliwe jest na przykład przetwarzanie tekstu, a następnie rozpoznawanie obrazów przy wykorzystaniu fizycznie tej samej części zasobów FPGA.

2 Przedmiot badań

W rozprawie duża uwaga przywiązywana jest do efektywnej *sprzętowej realizacji* algorytmów programowania dynamicznego wykorzystywanych w metodach analizy danych tekstowych. Należy wyjaśnić, że pod pojęciem „realizacji sprzętowej“ algorytmu rozumiane jest wykonanie algorytmu przy użyciu układu logicznego (zbiór bramek logicznych) lub obwodu elektronicznego, nie zaś tradycyjnego programu komputerowego uruchomionego na procesorze ogólnego przeznaczenia (CPU). W przypadku rozprawy wybrane algorytmy zrealizowane są przez odpowiednio połączone bramki logiczne w blokach układu cyfrowego FPGA z wykorzystaniem języka opisu sprzętu VHDL (ang. *Very High Speed Integrated Circuit Hardware Description Language*). Chcąc dokonać sprzętowej realizacji pewnych algorytmów trzeba zmierzyć się nie tylko z zagadnieniami zrównoleglania pętli czy ograniczeniami zasobów, ale przede wszystkim z problemami wynikającymi z utraty stabilności numerycznej (Pietras 2015, Gupta et al. 2015b, Hulden 2012). Obliczenia stosowane w różnych podejściach probabilistycznych (np. w modelach Markowa, sieciach bayesowskich) mają tendencję do utraty stabilności głównie ze względu na rodzaj wykonywanych obliczeń oraz własności maszynowych reprezentacji liczb (w szczególności reprezentacji wg standardu IEEE 754).

Zaprezentowane metody, wykorzystujące ukryte modele Markowa (ang. *Hidden Markov Models*), pozwalają na pozyskiwanie informacji ze źródeł internetowych w taki sposób, że dla podanego na wejściu adresu URL (ang. *Uniform Resource Locator*) strony internetowej analizowany jest jej kod HTML, w wyniku czego **rozpoznawana jest** najpierw **treść właściwa** zawartego na stronie artykułu (czyli odfiltrowane zostają elementy graficzne strony, reklamy, komentarze, forum dyskusyjne, podpisy rysunków, itp.), następnie **przeprowadzana jest syntaktyczna analiza** tej treści (oznaczanie częściami mowy i częściami zdania), i wreszcie dla wybranych istotnych informacji **tworzony jest opis na różnych poziomach semantyki leksykalnej**.

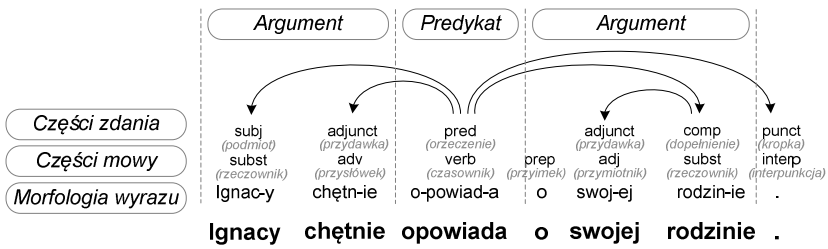
W rozprawie jako teoretyczną podstawę do reprezentowania semantyki tekstu zaproponowano autorską warstwową strukturę ukrytych modeli Markowa, gdzie na kolejnych poziomach

[2]Często etapy ekstrakcji i selekcji cech osadza się w konwolucyjnych sieciach neuronowych (czyli opartych na operacji mnożenia splotowego znanego z dziedziny przetwarzania obrazów), gdzie każdy neuron stanowi pewien filtr cech, stąd wiele cech przekłada się na dużą liczbę neuronów.

[3]Rozróżnia się rekonfigurację całościową i częściową układu FPGA oraz rekonfigurację statyczną (kiedy układ jest nieaktywny) i dynamiczną (podczas pracy układu) (Aithal 2016).

przetwarzania odbywa się: pozyskiwanie treści właściwej, a następnie anotacja tej treści przypisująca wyrazom kategorie syntaktyczne i dalej kategorie zależnościowe. Tym samym analiza na wyższym poziomie posiłkuje się informacjami uzyskanymi z analizy na niższym poziomie przetwarzania. Model ten znajduje zastosowanie w wielu przypadkach przetwarzania tekstów języka naturalnego, w tym opisu semantycznego. Jako autorskie elementy opracowanego rozwiązania należy wskazać przede wszystkim: zaprojektowanie modeli HMM (czyli odpowiedni dla realizowanych zadań dobór przestrzeni stanów i obserwacji), a także propozycję technik prowadzenia obliczeń w ramach tych modeli z zachowaniem stabilności numerycznej (w szczególności przy ograniczonej precyzji zmiennoprzecinkowej). Dzięki powyższym możliwa jest dalsza – semantyczna – analiza tekstu, która nie zawiera już elementów nowatorskich i jest przeprowadzana z wykorzystaniem zasobów Słowosieci (Maziarz et al. 2016). Na tym etapie odbywa się identyfikacja funkcji semantycznej (Fillmore 1968), a w szczególności badanie pola semantycznego (Gao & Xu 2013) wybranych wyrazów. Gromadzony jest opis (synonimy, asocjacje, cechy i działania podmiotu, działania wobec podmiotu) z uwzględnieniem różnych relacji semantyki leksykalnej takich jak np.: hiperonimia, hiponimia i kohiponimia.

Rysunek 1, wzorowany na diagramach z prac (Wróblewska & Przepiórkowski 2014, str. 27) i (Przepiórkowski et al. 2012, str. 124), przedstawia poglądowo składniową strukturę przykładowego zdania *„Ignacy chętnie opowiada o swojej rodzinie“* z uwzględnieniem zależności i morfologii wyrazów. W typowym prostym zdaniu predykat jest zwykle czasownikiem, natomiast argument reprezentuje fraza odrzeczownikowa.



Rysunek 1: Składniowa struktura zdania z uwzględnieniem morfologii wyrazów.
(Źródło: *opracowanie własne*).

Warto zauważyć, że wiele metod określania funkcji semantycznej korzysta z analizy syntaktycznej. Wykazano, że jakość wyników analizy syntaktycznej ma duży wpływ na prawidłowe określenie funkcji semantycznej (Punyakanok et al. 2008, Ge & Mooney 2005). W pracy (Meza-Ruiz & Riedel 2009) wykorzystano model, oparty na logice Markowa, do identyfikacji słów bazowych, argumentów i predykatów. Istnieją również modele statystyczne używane do wspólnej oceny całej semantycznej struktury zdania. Model opisany w pracy (Cohn & Blunsom 2005), oparty na warunkowych polach losowych, dokonuje identyfikacji argumentów i przypisania im funkcji semantycznych. W pracy (Abney & Light 1999) do określenia predykatów i ich argumentów wykorzystano ukryty model Markowa.

Metody uczenia maszynowego z pełnym nadzorem potwierdziły w wielu różnych opracowaniach swoją skuteczność w wykonywaniu zadań związanych z rozpoznawaniem funkcji semantycznej. Prowadzenie prac dotyczących zastosowania metod uczenia maszynowego w celu określania semantyki dla tekstów w języku polskim jest utrudnione przez brak odpowiednio dużych oznaczonych korpusów, które mogłyby stanowić materiał uczący. Stąd i badań poświęconych problemowi automatycznego określania struktury semantycznej w tekstach w języku polskim jest niewiele. Do rozwoju tej dziedziny przyczynili się twórcy zasobów Składnicy Treebank (Hajnicz 2014) jak i Słowosieci (Maziarz et al. 2016), które to zasoby wykorzystane zostały również w rozprawie do maszynowego nauczenia modeli i analizy semantycznej tekstów w języku polskim.

3 Metody naukowe stosowane podczas wykonywania badań

We wczesnej fazie prac badawczych zastosowana została metoda intuicyjna. Badane zjawiska utraty dokładności towarzyszące sprzętowej realizacji algorytmów o zredukowanej reprezentacji zmiennoprzecinkowej wymagały wyjaśnienia, wskazania założeń, ustalenia hipotez oraz wyboru metody roboczej. Spośród metod stosowanych w trakcie prowadzania badań wyróżnić należy poniższe.

- Metoda obserwacyjna
  - Obserwacja wpływu wielkości skrajnych prawdopodobieństw występujących w modelu HMM oraz reprezentacji zmiennoprzecinkowej na utratę stabilności numerycznej przy wykonywaniu algorytmów ukrytych modeli Markowa podczas badania długich sekwencji obserwacji.
- Metoda eksperymentalna
  - Eksperymenty z grupowaniem informacji z przestrzeni tekstowej i międzytekstowej dla danych zapisanych w formacie HTML.
  - Eksperymenty z zastosowaniem metod półautomatycznego nauczania ukrytych modeli Markowa w celu pozyskiwania informacji z danych zapisanych w formacie HTML.
  - Eksperymenty z doborem przestrzeni stanów w ukrytych modelach Markowa.
  - Eksperymenty z doborem gramowości w n-gramowej obserwacji w ukrytym modelu Markowa.
  - Eksperymenty z zastosowaniem metod wygładzania przy maszynowym uczeniu ukrytych modeli Markowa.
  - Eksperymenty obliczeniowe algorytmów przetwarzania ukrytych modeli Markowa z redukcją precyzji reprezentacji liczb zmiennoprzecinkowych.
  - Eksperymenty rearanżacji algorytmów przetwarzania ukrytych modeli Markowa w celu ich efektywnej implementacji sprzętowej.
- Metoda statystyczna
  - Szacowanie wartości parametrów w ukrytych modelach Markowa.
  - Określenie jakości odwzorowania wyrazów przy wykorzystaniu afiksów brzegowych w obserwacji w ukrytym modelu Markowa.
  - Określenie jakości detekcji za pomocą miar precyzji dla modeli HMM przeznaczonych do pozyskiwania informacji ze źródeł HTML.
  - Określenie jakości detekcji za pomocą miar precyzji, czułości i F1 dla modeli HMM przeznaczonych do rozpoznawania części mowy w zdaniu.
  - Określenie jakości detekcji za pomocą miar precyzji, czułości i F1 dla modeli HMM przeznaczonych do rozpoznawania typów relacji zależnościowych w zdaniu.
  - Porównanie uzyskanych wyników z wynikami raportowanymi w innych pracach naukowych.
- Metoda indywidualnych przypadków
  - Anotacja częściami mowy wybranych zdań.
  - Anotacja typów relacji zależnościowych w wybranych zdaniach.
  - Opis informacji na rożnych poziomach semantyki leksykalnej na podstawie analizowanego tekstu.

4 Główny cel rozprawy

Głównym celem rozprawy jest zaprojektowanie ukrytych modeli Markowa, które pozwolą na analizę danych tekstowych poprzez algorytmy programowania dynamicznego realizowane sprzętowo.

W powyższym celu głównym mieszczą się następujące **cele szczegółowe**:

1. opracowanie metody pozyskiwania istotnych danych tekstowych ze źródeł HTML,
2. opracowanie metody oznaczania tekstu częściami mowy i częściami zdania,
3. opracowanie metody opisu istotnych informacji w tekście na różnych poziomach semantyki leksykalnej,
4. sprzętowa realizacja modeli Markowa z możliwością redukcji precyzji zmiennoprzecinkowej przy zachowaniu założonej dokładności.

Odnośnie drugiego punktu należy zaznaczyć, że oznaczanie tekstu częściami mowy (np. rzeczownik, czasownik, przymiotnik) i częściami zdania (np. podmiot, orzeczenie) nazywane jest bardziej formalnie odpowiednio anotacją morfosyntaktyczną oraz anotacją typami relacji zależnościowych. **Teza** pracy jest sformułowana następująco:
Sprzętowe realizacje ukrytych modeli Markowa o zredukowanej reprezentacji zmiennoprzecinkowej pozwalają na analizę sekwencji tekstowych z zadowalającą dokładnością wykorzystując logarytmowane wersje algorytmów Viterbiego oraz Forward-Backward.

5 Zadania do wykonania

1. Opracowanie metody pozyskiwania istotnych danych tekstowych ze źródeł HTML:
   - Opracowanie niezależnego od języka grupowania obserwacji (sposób binarny i wyrażeniowy) dla przestrzeni międzytekstowych w źródłach HTML.
   - Określenie kategorii semantycznych dla stanów w modelu HMM do pozyskiwania istotnych danych ze źródeł HTML.
   - Realizacja edytora wizualnego do sprawnego oznakowywania treści w aplikacji HMM-Toolbox.
   - Utworzenie bazy danych uczących i testowych na podstawie artykułów z portali informacyjnych.
   - Opracowanie trzyfazowego nauczania modelu HMM (wstępne szacowanie parametrów, ręczna korekta, końcowa reestymacja w wariancie treningu Viterbiego).
   - Wyprodukowanie ukrytych modeli Markowa dedykowanych do pozyskiwania istotnych danych tekstowych ze źródeł HTML.
2. Opracowanie metody oznaczania tekstu częściami mowy i częściami zdania:
   - Opracowanie jednolitego stylu agregacji n-gramowej obserwacji na podstawie morfemów brzegowych z możliwością stopniowego wygaszania wcześniejszych afiksów.
   - Utworzenie bazy danych uczących i testowych na podstawie banków drzew składniowych i zależnościowych dla języka polskiego i angielskiego.
   - Opracowanie dwuwarstwowej struktury z modelami HMM, gdzie przestrzeń obserwacji na wyższym poziomie zawiera informacje zarówno z afiksów wyrazów, jak i informacje o części mowy odkrytej przez model Markowa z warstwy niższej.
   - Wyprodukowanie ukrytych modeli Markowa dedykowanych do rozpoznawania w zdaniach części mowy, grup frazowych i typów relacji zależnościowych dla języka polskiego i angielskiego.

3. Opracowanie metody opisu istotnych informacji w tekście na różnych poziomach semantyki leksykalnej:
    - Disambiguacja znaczenia wyrazów przez odkrywanie części mowy dedykowanym modelem HMM.
    - Procedura rozmytego dopasowania (na podstawie zmodyfikowanego algorytmu Levenshteina) wyrazów z łagodzeniem wpływu końcówki fleksyjnej przy koniugacji i deklinacji wyrazów w języku polskim.
    - Migracja elementu leksykalnego. Uzyskany opis semantyczny na wyższym poziomie semantyki leksykalnej.
4. Sprzętowa realizacja modeli Markowa z możliwością redukcji precyzji zmiennoprzecinkowej przy zachowaniu założonej dokładności:
    - Opracowanie lematu dla określenia numerycznie bezpiecznej długości badanej sekwencji obserwacji przy uwzględnieniu zredukowanej precyzji reprezentacji liczb zmiennoprzecinkowych.
    - Opracowanie wydajnej sprzętowej jednostki wielu aproksymacji przekształceń logarytmicznych i wykładniczych.
    - Realizacja algorytmu Bauma-Welcha dla sprzętowego zrównoleglenia pętli.
    - Przyspieszenie algorytmu Viterbiego przez zrównoleglenie obliczeń w przód i wykonanie nawracania kawałkami.

6 Wartość teoretyczna

Jak powszechnie wiadomo (Rabiner 1989), obliczenia w ramach HMM mogą stać się kłopotliwe, zwłaszcza w przypadku długich sekwencji. Nawet jeśli obliczenia te są wykonywane w przestrzeni logarytmicznej, należy pamiętać, że istnieje zawsze pewna granica długości badanej sekwencji, poza którą obliczenia mogą stać się numerycznie niestabilne. Zjawisko to należy rozpatrywać w parze z rozmiarem (szerokością bitową) reprezentacji liczb zmiennoprzecinkowych. W celu implementacji FPGA w dalszej części pracy zastosowana zostanie redukcja standardowej reprezentacji o podwójnej precyzji (64 bity) do precyzji: pojedynczej (32 bity), połówkowej (16 bitów) lub ćwiartkowej (8 bitów). Przedstawiony w rozprawie lemat stanowi główny teoretyczny wynik pracy. Pokazuje on ilościowo, w jaki sposób można ustalić długość najkrótszej sekwencji niebezpiecznej numerycznie w terminach skrajnych prawdopodobieństw występujących w modelu HMM oraz przy pomocy zadanej szerokości bitowej mantysy dla reprezentacji zmiennoprzecinkowej. Wynik ten został opublikowany w pracy (Pietras & Klęsk 2017).

7 Wartość praktyczna

Na potrzeby prac badawczych utworzona została w pełni samodzielna aplikacja HMM-Toolbox. Aplikacja ta zawiera niezbędne zestawy narzędzi programistycznych w postaci przygotowanych funkcji i algorytmów oraz graficzny interfejs użytkownika usprawniający realizację poszczególnych zadań projektowych. Za pomocą HMM-Toolbox wyprodukowane (utworzone, skonfigurowane, nauczone i zweryfikowane) zostały liczne ukryte modele Markowa:

- Modele HMM przeznaczone do pozyskiwania informacji z źródeł HTML:
    - model pozyskiwania treści artykułów z witryny www.onet.pl,
    - model pozyskiwania treści artykułów z witryny www.theguardian.com.
- Modele HMM przeznaczone do rozpoznawania części mowy w zdaniu:
    - model do rozpoznawania części mowy dla języka angielskiego,

  - model do rozpoznawania podstawowych części mowy dla języka polskiego,
  - model do rozpoznawania rozszerzonych części mowy dla języka polskiego.

- Modele HMM przeznaczone do rozpoznawania typów relacji zależnościowych w zdaniu:
  - model do rozpoznawania grup frazowych dla języka angielskiego,
  - model do rozpoznawania typów relacji zależnościowych dla języka angielskiego,
  - model do rozpoznawania typów relacji zależnościowych dla języka polskiego.

Na potrzeby wydajnej realizacji sprzętowej opracowana została seria algorytmów dedykowanych pod implementacje FPGA projektowanych metodą "dziel i zwyciężaj". Bloki algorytmiczne utworzone przez syntez wysokiego poziomu:

- Moduł logarytmicznej wersji algorytmu Baum-Welch D&C (ze zintegrowanymi funkcjami obliczeniowymi algorytmów Forward-Backward D&C, Gamma-Xi D&C oraz Numerator-Denominator D&C).
- Moduł logarytmicznej wersji algorytmu Viterbi D&C z nawracaniem kawałkami.

Na potrzeby przyspieszenia rozszerzonej arytmetyki logarytmów zaprojektowana została (w opisie VHDL) specjalna jednostka umożliwiająca aproksymacje wielu przekształceń logarytmicznych i wykładniczych MLEAU (ang.*Multiple Logarithm Exponent Approximation Unit*)

- Projekt IP-Core dla MLEAU utworzony w języku opisu sprzętu VHDL. Do projektu przypisany jest testbench oraz pliki dla symulacji Matlab.

8 Akceptacja wyników przez społeczność naukową

Czasopisma i materiały konferencyjne, w których recenzenci oceniali wyniki badań:

- Information Systems in Management XVIII, (ISBN 978-83-7583-566-3), WULS Press Warsaw (2013).
- IEEE Xplore Digital Library, IEEE (2014).
- Przegląd Elektrotechniczny, Wydawnictwo Czasopism i Książek Technicznych SIGMA-NOT Spółka z o.o (2015).
- AIP Conference Proceedings 1648, AIP Publisching (2015).
- Hard and Soft Computing for Artificial Intelligence, Multimedia and Security, Springer (2017).
- Bulletin of the Polish Academy of Sciences: Technical Sciences (2017).

Konferencje, na których zaprezentowano wyniki badań:

- III Ogólnopolska Konferencja Naukowa ICT Young, Polska, Gdańsk (2013).
- XI Ogólnopolska Konferencja Naukowa Systemy Informacyjne w Zarządzaniu, Polska, Warszawa (2013).
- XXIV International Conference on Field Programmable Logic and Applications, Niemcy, Monachium (2014).

- XX International Multi-Conference on Advanced Computer Systems, Polska, Miedzyzdroje (2014).
- XII International Conference on Numerical Analysis and Applied Mathematics, Grecja, Rodos (2014).
- XIX International Multi-Conference on Advanced Computer Systems, Polska, Międzyzdroje (2016).

9 Struktura i układ pracy

Rozprawa składa się z ośmiu rozdziałów, bibliografii, oraz załącznika. Pełny zakres pracy składa się z 151 stron, 58 ilustracji oraz 36 tabel.

W rozdziale pierwszym przedstawione zostały wyzwania z zakresu lingwistyki komputerowej oraz przykłady zastosowania dedykowanych rozwiązań sprzętowych w celu akceleracji obliczeń związanych z algorytmami uczenia maszynowego. W ramach wprowadzenia do analizy syntaktycznej i semantycznej przedstawiono przegląd opracowanych w tym celu systemów. Ponadto pokrótce opisano sposoby przypisywania kategorii syntaktycznych i semantycznych wyrazom w analizowanym tekście.

Rozdział drugi zawiera przegląd algorytmów wykorzystywanych w obliczeniach ukrytych modeli Markowa. Przedstawiona została w nim ogólna struktura zarówno modelu, jak i wykorzystywanych obserwacji. Ze względu na rozważania dotyczące stabilności numerycznej stanowiące istotny element rozprawy, algorytmy związane z modelami Markowa zostały podane w tym rozdziale w dwóch wersjach: tradycyjnej (operującej na prawdopodobieństwach) i logarytmowanej (operującej na logarytmach prawdopodobieństw). Zaprezentowana również została metodologia nauczania i reestymacji[4] parametrów modeli, ponadto pokrótce przedstawiono aplikację HMM-Toolbox, utworzoną na potrzeby badań i eksperymentów przeprowadzonych w ramach rozprawy.

Trzeci rozdział dotyczy danych tekstowych pochodzących ze źródeł HTML. Opisane zostały w nim struktura języka HTML oraz sposoby reprezentacji danych. Przedstawiono historię i przegląd technologii pozyskiwania informacji. Zaprezentowana została autorska metoda (niezależna od języka) służąca do pozyskiwania istotnych danych tekstowych oraz rozpoznawania treści właściwej z wykorzystaniem ukrytych modeli Markowa. Obszernie opisana została także procedura przygotowania i nauczania modelu. Uzyskane wyniki kategoryzacji danych ze źródeł HTML podsumowują tę część pracy.

Czwarty rozdział poświęcony został analizie syntaktycznej i semantycznej. Przedstawiono w nim przegląd metod konstruowania drzew składniowych. Wykorzystując techniki nadzorowanego nauczania maszynowego, dokonane zostało rozpoznawanie kategorii syntaktycznych oraz typów relacji zależnościowych w tekstach w języku polskim, jak i w języku angielskim. Do wykonania tego zadania wykorzystano zbiory drzew składniowych, których opis zawarto w tym rozdziale (wraz z opisem zestawu rozpoznawanych kategorii syntaktycznych). Dodatkowo, z dużą szczegółowością przedstawione zostały także: n-gramowy model obserwacji zorientowanej na afiksy, sposoby konstruowania drzewa zależności oraz sposoby określania podstawowych funkcji w strukturze zdania. Z kolei na potrzeby analizy semantycznej utworzone zostały odrębne zestawy stanów modelu Markowa, opracowane na podstawie oznaczeń drzew zależnościowych „Składnica“ dla języka polskiego (Hajnicz 2014) i „PropBank“ dla języka angielskiego (Palmer et al. 2006). Wyczerpująco opisano procedurę nauczania warstwowych ukrytych modeli Markowa. W podsumowaniu rozdziału, wyniki otrzymane dla prezentowanych ukrytych modeli Markowa porównane zostały z wynikami raportowanymi w pracach naukowych: (Surdeanu et al. 2008), (Iwakura & Okamoto 2008) i (Wróblewska 2014, 2018)).

Rozdział piąty skupia się na opisie istotnych informacji, zawartych w tekście, na różnych poziomach semantyki leksykalnej. Pokrótce przedstawione zostały w nim główne cechy sieci

[4]Pod pojęciem „reestymacji“ rozumiane jest wielokrotne doszacowywanie parametrów modelu HMM na podstawie danych uczących zgodnie z procedurą Bauma–Welcha znaną pod nazwą „Baum–Welch reestimation“ (Leonard E. Baum & Weiss 1970).

semantycznej i poziomów semantyki leksykalnej. Zaprezentowano metody disambiguacji i dopasowywania wyrazów. Korzystając z zasobów Słowosieci (ang. *Wordnet*)[5] oraz ustrukturyzowanej formy informacji powstałej w wyniku analizy syntaktycznej i semantycznej opracowana została metoda wielopoziomowego opisu relacji semantycznych występujących w tekście.

Szósty rozdział poświęcono określeniu ram stabilności numerycznej obliczeń. Zaprezentowano, a jednocześnie udowodniano możliwość występowania warunków zapewniających bezpieczeństwo numeryczne ukrytych modeli Markowa o zredukowanej reprezentacji liczbowej pozwalającej na analizę sekwencji tekstowych z zadowalającą dokładnością, wykorzystując logarytmowane wersje algorytmów programowania dynamicznego.

W rozdziale siódmym zaprezentowana została sprzętowa akceleracja algorytmów programowania dynamicznego wykorzystywanych w obliczeniach ukrytych modeli Markowa. Korzystając z technik syntezy wysokiego poziomu utworzono w układzie FPGA architekturę obliczeniową o wysokim stopniu zrównoleglenia opartą o jednostki przetwarzania stanów. Przy sprzętowej realizacji algorytmów wykorzystano metodologię projektową „dziel i zwyciężaj". Dla wydajnego dekodowania ukrytych ścieżek w ramach modeli Markowa przedstawiono modyfikację algorytmu Viterbiego. Szczególna uwaga poświęcona została jednostce aproksymacji wielu funkcji logarytmicznych i wykładniczych „MLEAU", która pozwala na wydajne obliczenia w dziedzinie logarytmicznej.

Pracę kończy podsumowanie zawierające wnioski na temat dokładności oraz skuteczności opracowanych metod.

[5]Wordnet oraz Słowosieć to fizycznie różne (choć odpowiadające sobie) zasoby. Warto zaznaczyć, że twórcy powstałych później zasobów polskich – Slowosieci – również posługują się angielskim tłumaczeniem „Wordnet". W związku z powyższym w rozprawie te terminy używane są zamiennie.

10 Zawartość pracy

10.1 Aplikacja HMM-Toolbox

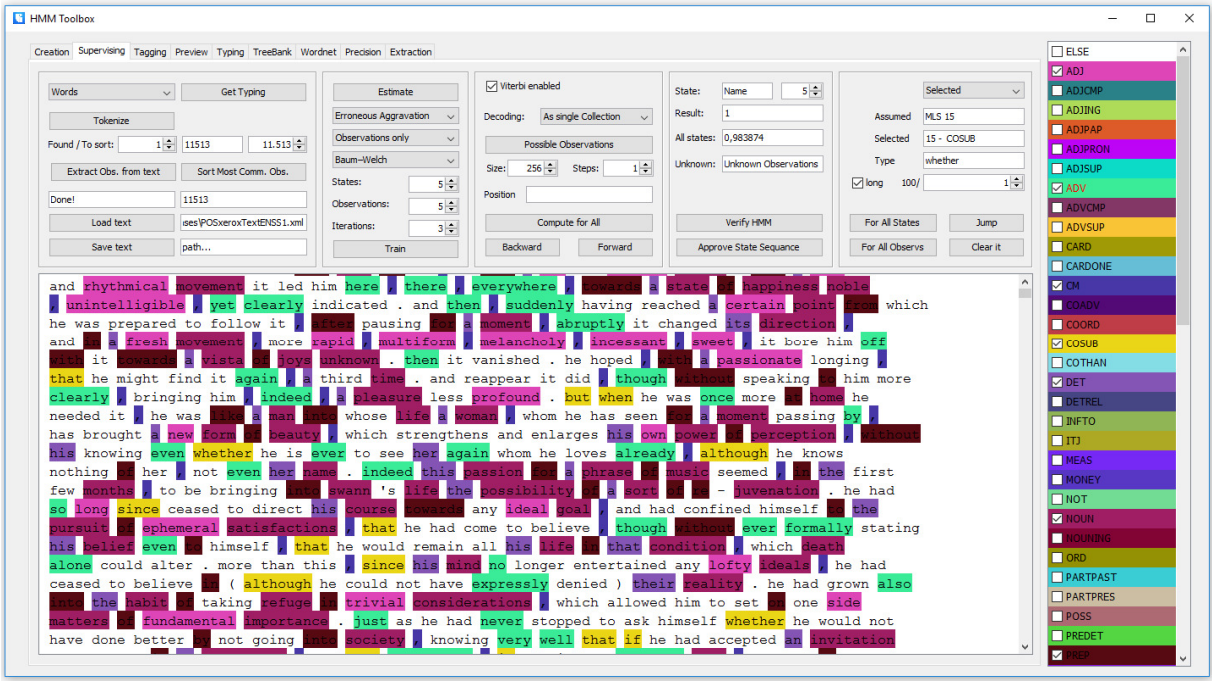
Na potrzeby prac badawczych utworzona została w pełni samodzielna aplikacja zawierająca niezbędne zestawy narzędzi programistycznych w postaci przygotowanych funkcji i algorytmów oraz graficzny interfejs użytkownika usprawniający realizację poszczególnych zadań projektowych. Struktura tworzonego modelu jest w pełni konfigurowalna. Dodatkowe informacje opisujące model również są dostępne i edytowalne z poziomu GUI. Ponadto HMM-Toolbox obsługuje n-gramowy konfigurator obserwacji pozwalający na zdefiniowanie długości afiksów i prefiksów oraz rzędu gramowości.

Kolejne zakładki aplikacji HMM-Toolbox prezentowane w rozprawie odpowiadają typowemu przebiegowi pracy użytkownika z aplikacją:

- tworzenie modelu (zakładka *Creation*),
- uczenie nadzorowane modelu (zakładka *Supervising*),
- graficzna wizualizacja struktury drzewa (zakładka *Treebank*),
- zbiorczego oznaczania treści (zakładka *Tagging*),
- podgląd pozyskiwanej informacji (zakładka *Supervising*),
- wpisywanie i rozpoznawanie nowego tekstu (zakładka *Typing*),
- opis informacji na różnych poziomach semantyki (zakładka *Wordnet*),
- redukcja reprezentacji liczbowej (zakładka *Precision*).

Ponadto w zakładce *Extraction* znajdują się przydatne funkcje importu, eksportu i konwersji różnych struktur danych wykorzystanych podczas realizacji prac badawczych.

Rysunek 2 przedstawia HMM-Toolbox w trybie nadzorowania. W głównym oknie aplikacji widoczny jest tekst w języku angielskim. Dla każdego wyrazu określona jest pewna kategoria znaczenia syntaktycznego (przypisana do stanu w modelu), co jest zobrazowane przez podświetlenie wyrazu odpowiednim kolorem.



Rysunek 2: Nauczanie i weryfikacja modelu w HMM-Toolbox.
(Źródło: *opracowanie własne*).

Aplikacja HMM-Toolbox, ze względu na rozbudowane funkcje, pomocna jest również w celach dydaktycznych szczególnie podczas nauczania, weryfikacji i wizualizacji ukrytych modeli Markowa.

10.2 Pozyskiwanie informacji z danych w formacie HTML

Zaproponowane zostały ukryte modele Markowa, dokonujące klasyfikacji danych tekstowych zapisanych w formacie HTML. Ponadto, dla danego ukrytego modelu Markowa możliwe jest określenie jakie byłoby prawdopodobieństwo wygenerowania badanej ścieżki obserwacji. Prawdopodobieństwo to można porównywać między poszczególnymi modelami w celu znalezienia modelu dokonującego najtrafniejszej kategoryzacji danych z formatu HTML. Warto podkreślić, że zaprojektowany system informatyczny, ma pozwolić użytkownikowi bez specjalnego przeszkolenia (nie będącemu programistą) na odpowiednie oznakowanie treści witryny internetowej i na tej podstawie zostanie wygenerowany model.

W wyniku wstępnego sparsowania źródła HTML otrzymywana jest sekwencja tokenów, która to poddana zostaje odpowiedniemu podziałowi na zbiory tokenów, występujące w osobnych przestrzeniach międzytekstowych. Na tym etapie wstępny parser przypisuje napotkanym ciągom znaków pasującą kategorię. Tak więc w tokenie przechowywana jest jedynie informacja o wykrytej kategorii i rodzaju znacznika, reszta informacji (ciąg znaków, zawartość znacznika, itp.) jest tracona. W ten sposób unikane są niejednoznaczności i błędy występujące w formacie HTML.

10.2.1 Przestrzeń międzytekstowa w formacie HTML

Tekstem zapisanym w języku naturalnym, występujący pomiędzy znacznikami w formacie HTML nazywać będziemy przestrzenią tekstową, która oddziela kolejne przedziały przestrzeni międzytekstowej. Wystąpienie każdego przedziału przypisane jest do dyskretnej chwili czasu t, ponadto przyjmuje się, że dane zapisane w formacie HTML rozpatrywane są chronologicznie od góry do dołu; pierwszy rozpoznany zbiór występuje w chwili $t = 1$, a ostatni $t = T$.
W rozprawie przyjęto, że wystąpienie znaczników jest cechą charakterystyczną dla danego przedziału, dlatego zbiór znaczników w przestrzeni międzytekstowej może być użyty jako obserwacja (w rozumieniu obserwacji HMM). Grupowanie znaczników w danym przedziale rozpatrywana jest na dwa sposoby:

- **binarne** – na zasadzie czy dany znacznik w ogóle wystąpił w danym przedziale międzytekstowym, bez względu na kolejność czy liczność wystąpień;
- **w wyrażenia** – znaczniki występujące w przedziale międzytekstowym tworzą unikatowe wyrażenia, z uwzględnieniem ich kolejności, jak i liczności wystąpień.

Oba warianty poddane zostały ocenie w celu określenia, który z nich statystycznie lepiej grupuje istotne informacje dla zadania pozyskiwania informacji.

10.2.2 Przygotowanie modelu HMM

W celu pozyskania treści właściwej (takiej jak temat, tekst artykułu, autorstwo itp.) z portali informacyjnych przygotowany został ukryty model Markowa. Kategorie utożsamiane ze stanami w ukrytym modelu Markowa przedstawia Tabela 1.

Numer	Nazwa stanu	Opis
1	Inny	elementy nieoznakowane
2	Link	hiperłącza i odnośniki do innych stron
3	Menu	elementy umożliwiające poruszanie się na stronie internetowej
4	Element szkieletowy	powtarzalne elementy tj, przyciski, etykiety, ramki, itd.
5	Temat	temat artykułu
6	Podtemat	temat rozdziału w artykule
7	Tekst właściwy	tekst właściwy artykułu
8	Autor	autor artykułu
9	Obraz	występujące obrazy i grafiki
10	Tytuł	tytuł strony internetowej
11	Miejsce i Czas	czasoprzestrzenna informacja odnośnie artykułu
12	Opis	opis treści zawartej w artykule
13	Komentarz	komentarze pozostawione przez użytkowników
14	Reklama	każda wykryta forma reklamy
15	Udostępnianie	elementy umożliwiające udostępnianie artykułu w innych serwisach
16	Koniec strony	stan ważny z punktu widzenia nauczania modelu

Tabela 1: Zestaw kategorii przypisanych do stanów modelu.
(Źródło: *opracowanie własne*).

Prezentowany w Tabeli 1 zbiór stanów określony został w sposób empiryczny na podstawie prac eksperymentalnych z wykorzystaniem różnych (mniejszych, większych) konfiguracji zbiorów.

10.2.3 Wyniki dla pozyskiwania informacji

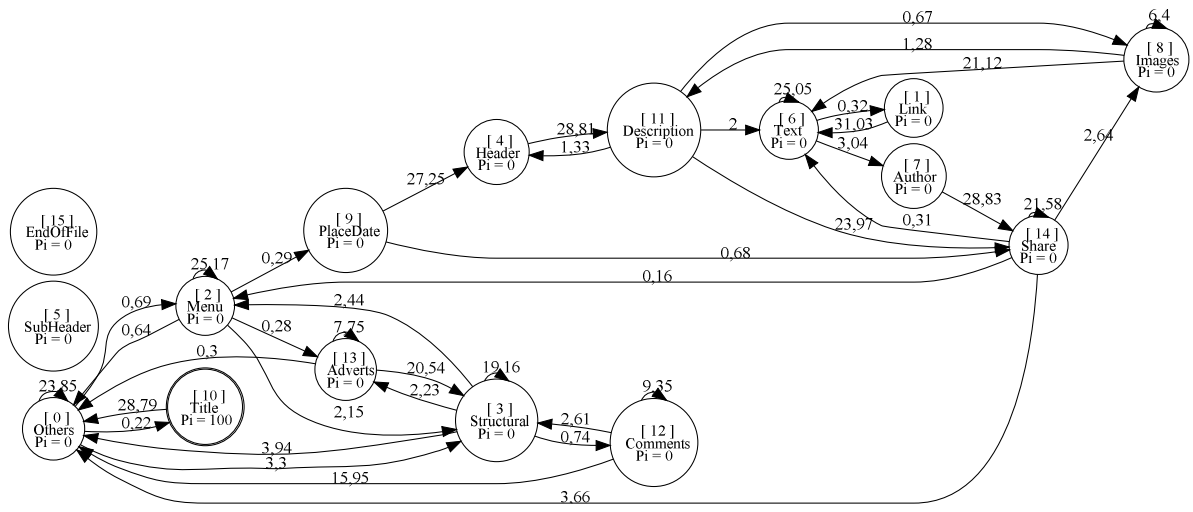
Eksperymenty z prototypem systemu wykazały możliwość praktycznego zastosowania proponowanej metody pozyskiwania informacji. Tabela 2 przedstawia wyniki precyzji dla pozyskiwania informacji z witryn portalu internetowego *www.onet.pl* z wykorzystaniem obserwacji grupowanej **binarnie**. W wyniku uczenia określonych zostało 341 różnych obserwacji binarnych (a więc mniej niż dla obserwacji wyrażeniowych). Ocena modelu na zestawie testowym wykazała, że tylko 3% (tj. 185) spośród ponad 5 tysięcy przestrzeni międzytekstowych nie udało się zidentyfikować, stąd otrzymały one kategorię obserwacji „nieznana“. Precyzja identyfikacji dla obserwacji nieznanych wyniosła 74.42% natomiast dla znanych obserwacji precyzja wyniosła już 99.54%, tym samym całkowita precyzja klasyfikacji dla wszystkich obserwacji wyniosła 99.09%.

Numer	Nazwa stanu	Precyzja dla znanych [%]	Precyzja dla nieznanych [%]	Precyzja całkowita [%]
1	Inne	97.69	76.24	96.45
2	Link	100	-	100
3	Menu	98.80	35.09	95.07
4	Element szkieletowy	98.72	86.36	98.54

5	Temat	100	-	100
6	Podtemat	100	-	100
7	Tekst właściwy	98.85	100	98.86
8	Autor	100	-	100
9	Obraz	100	-	100
10	Miejsce i czas	100	-	100
11	Tytuł	100	-	100
12	Opis	100	-	100
13	Komentarz	98.97	0.00	97.96
14	Reklama	99.68	-	99.68
15	Udostępnienie	100	-	98.95
16	Koniec strony	100	-	100
Razem		99.54	74.42	99.09

Tabela 2: Precyzja klasyfikacji informacji z artykułów portalu *www.onet.pl* z wykorzystaniem obserwacji grupowanej binarnie.
(Źródło: *opracowanie własne*).

Rysunek 3 ukazuje model Markowa uzyskany dla portalu *www.onet.pl*. Widoczne na rysunku przejścia między stanami mają określony kierunek wyrażony strzałką oraz określone prawdopodobieństwo wyrażone wartością procentową. W każdym stanie widoczne jest początkowe prawdopodobieństwo wystąpienia oznaczone jako Pi. Stany pomocnicze (np. EndOfFile) oraz stany które nie wystąpiły w procesie uczenia są w modelu nieosiągalne i przedstawione jako niepołączone.



Rysunek 3: Model Markowa dla klasyfikacji artykułów z portalu *www.onet.pl*.
(Źródło: *opracowanie własne*).

Wszystkie prezentowane w rozprawie grafy wizualizujące modele Markowa zostały utworzone z użyciem oprogramowania Graphviz (Ellson et al. 2001) bazując na opisie struktury (w formacie DOT) nauczonych modeli wygenerowanym przez aplikacje HMM-Toolbox.

10.3 Syntaktyczna analiza treści

W rozprawie analiza syntaktyczna przeprowadzona jest zarówno dla języka angielskiego, jak i polskiego, stąd prezentowane są tu dwa banki drzew: Penn Treebank dla języka angielskiego oraz Składnica frazowa dla języka polskiego.

10.3.1 Wybór modeli i danych uczących

W trakcie prowadzenia badań sprawdzone zostały różne kombinacje oznakowanych syntaktycznie tekstów w celu wyłonienia najkorzystniejszego podejścia. Dla języka angielskiego niewątpliwie najkorzystniej jest przeprowadzić uczenie modelu w oparciu o dobrze znany bank drzew Penn Treebank, który to ma ugruntowaną pozycję w dziadzinie lingwistyki komputerowej. Struktura banku zawiera oznakowanie danych tekstowych na rożnym poziomie syntaktyczno–semantycznym, począwszy od oznakowaniu tekstu częściami mowy i częściami zdania, a kończąc na określeniu funkcji semantycznych. Dla języka polskiego wybrany został bank drzew Składnica, będący największym zbiorem oznakowanych syntaktycznie i semantycznie zdań w języku polskim. Ponadto, zaletą obu banków drzew jest ich szerokie rozpowszechnienie więc i możliwość porównania otrzymywanych wyników rozpoznawania.

W wynikach ilościowych raportowanych w dalszych częściach tego rozdziału dokładność działania poszczególnych modeli Markowa oceniona została za pomocą miar: precyzji, czułości oraz F1. Przy czym każda z tych miar była wyznaczana na rzecz różnych klas rozpoznawanych w ramach danego typu zadania. Klasy te są utożsamiane z odpowiednimi stanami w modelach Markowa.

10.3.2 Model rozpoznawania kategorii syntaktycznych dla języka polskiego

Kategorie syntaktyczne dla języka polskiego przypisane do stanów w ukrytym modelu Markowa określone w (Przepiórkowski et al. 2012) jako klasy fleksemów przedstawia Tabela 3.

Numer stanu	Nazwa stanu	Opis	Przykłady językowe
0	unknown	Kategoria nieznana	{}
1	subst	Rzeczownik	*„kot“, „profesorowie“, „godziny“, „komitetu“, „pieniądze“*
2	praet	Czasownik pseudoimiesłowowy	*„jadał“, „stał“, „miał“, „zranił“, „doprowadziło“*
3	qub	Kublik (partykuła)	*„nie“, „jednak“, „również“, „się“, „chociaż“*
4	prep	Przyimek	*„pod“, „w“, „do“, „przez“, „o“, „na“*
5	adj	Przymiotnik	*„polski“, „czerwony“, „fałszywej“, „podstawowe“, „młodzi“*
6	interp	Interpunkcja	*„;“, „(“, „]“*
7	punct	Koniec zdania	*„.“*
8	fin	Czasownik forma nieprzeszła	*„jadam“, „jesteś“, „przepraszam“, „zjawia“, „płynie“*
9	ppron3	Zaimek trzecioosobowy	*„on“, „jemu“, „jej“, „jego“, „go“*
10	conj	Spójnik współrzędny	*„oraz“, „lub“, „lecz“, „i“, „jednak“*
11	ger	Czasownik odsłownik (rzeczownik odczasownikowy, imiesłów odczasownikowy)	*„jadanie“, „malowanie“, „zwiedzanie“, „dokonaniu“, „wykorzystywanie,“*
12	ppas	Imiesłów przymiotnikowy bierny	*„jadany“, „zaprezentowana“, „zgubionym“, „złożony,“*
13	adv	Przysłówek	*„bardziej“, „kiedy“, „obecnie“, „osobiście“, „dłużej“*

14	impt	Czasownik rozkaźnik (wykrzyknik)	*„jadaj“, „porzucaj“, „wyglądaj“, „czyn“, „pytaj“*
15	comp	Spójnik podrzędny	*„że“, „aby“, „jednak“, „choć“, „jeśli“, „żeby“*
16	inf	Czasownik bezokolicznik	*„być“, „składać“, „jadać“, „znaleźć“, „być“, „marzyć“*
17	Bedzie	Forma przyszła czasownika być	*„będę“*
18	pred	Predykatyw	*„trzeba“, „słychać“, „można“, „to“, „dość“*
19	:	Dwukropek	*„:“*
20	brev	Skrót	*„dr“, „np“, „zł“, „tys“*
21	dot	Wylicznik	*„...“, „tys.“, „r.“, „proc.“*
22	num	Liczebnik główny	*„sześć“, „dużo“, „wiele“, „sporo“, „5“, „obie“*
23	pcon	Imiesłów przysłówkowy współczesny	*„jadając“, „rewanżując“, „biorąc“, „tracąc“, „notując“*
24	ppron12	Zaimek nietrzecioosobowy	*„ja“, „tobie“, „nas“, „mnie“*
25	imps	Czasownik bezosobowy	*„jadano“, „przeprowadzono“, „wyryto“, „postawiono“*
26	pact	Imiesłów przymiotnikowy czynny	*„jadający“, „napływające“, „kierujący“, „umacniającym“, „wygrzewający“*
27	aglt	Aglutynant czasownika być	*„chciał –em“, „by –m“, „sprawdzili -śmy, “*
28	interj	Wykrzyknik fatyczny (emotikon)	*„ach“, „psiakrew“, „o kurczę“, „precz“*
29	numcol	Liczebnik zbiorowy	*„sześcioro“, „trojga“, „dwojgiem“*
30	siebie	Siebie	*„sobą“, „siebie“*
31	adjc	Przymiotnik predykatywny	*„wesół“, „pewien“, „gotów“*
32	winien	Czasownik typu winien (forma teraźniejsza)	*„winna“, „powinni“*
33	adja	Przymiotnik przy-przymiotnikowy	*„polsko-ukraińska“, „chrześcijańsko-narodowy“, „warmińsko-mazurski“*
34	adjp	Przymiotnik po-przyimkowy	*„po polsku“, „po angielsku“, „od dawna“, „po prostu“, „po cichu“*
35	depr	Rzeczownik forma deprecjatywna	*„profesory“, „słoiki“, „chłopaki“, „wykształciuchy“*
36	burk	Burkinostka	*„omacku“, „trochu“, „na jaw“, „z dala“, „na oścież“*
37	xxx	Ciało obce	*„errare“, „humanum“, „however“*
38	-)	Emotikon	*„:-)“, „;-)“, „:d“, „:p, “*
39	pant	Imiesłów przysłówkowy przeszły!	*„wyprowadziwszy“, „zjadłszy“, „poznawszy“, „stwierdziwszy“*

Tabela 3: Kategorie syntaktyczne dla języka polskiego przypisane do stanów w ukrytym modelu Markowa.
(Źródło: *opracowane na podstawie (Przepiórkowski et al. 2012)*).

10.3.3 Wyniki rozpoznawania kategorii syntaktycznych dla języka polskiego

W wyniku uczenia określonych zostało 43 199 różnych obserwacji. Ocena modelu na zestawie testowym wykazała, że dla około 13.92% (tj. 9 969 spośród 73 945) wyrażeń nie udało się zidentyfikować pasującej obserwacji, stąd zostały one przypisane do kategorii „nieznana“. Precyzja rozpoznawania obserwacji nieznanych wyniosła 52.94%, natomiast dla znanych obserwacji precyzja wyniosła już 95.05%, tym samym całkowita precyzja klasyfikacji dla wszystkich obserwacji w zestawie testowym wyniosła 92.47%. Całkowita czułość klasyfikacji dla wszystkich obserwacji w zestawie testowym wyniosła 92.23%. Stąd, miara F1 dla przedstawionego klasyfikatora wyniosła 92.97%.

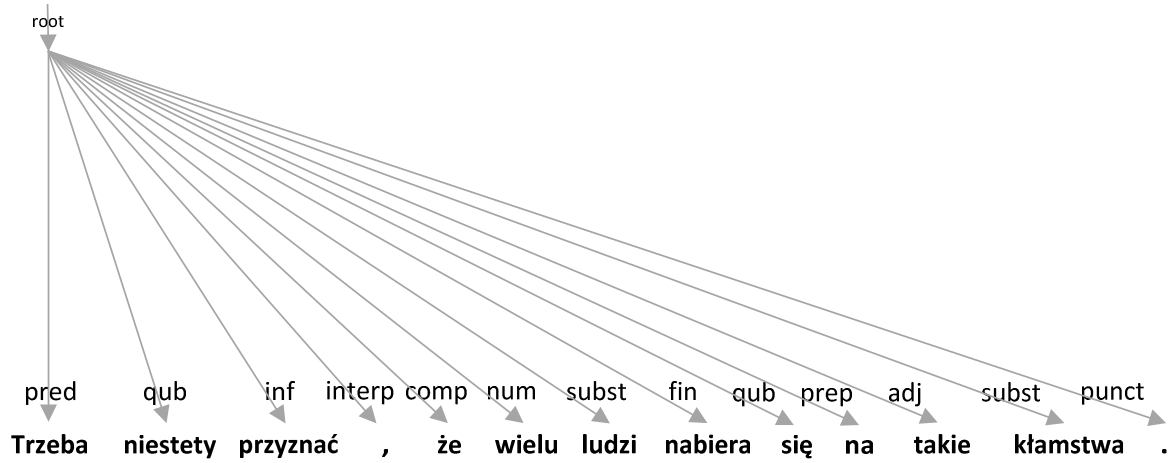
Numer stanu	Nazwa stanu	Precyzja dla znanych [%]	Precyzja dla nieznanych [%]	Precyzja wszystkich [%]	Czułość wszystkich [%]	Miara F1 wszystkich [%]
0	unknown	-	-	-	-	-
1	subst	97.03	50.04	79.10	97.34	87.28
2	praet	99.21	98.53	99.19	99.81	99.45
3	qub	95.61	-	95.61	91.04	93.27
4	prep	98.92	-	98.92	98.59	98.75
5	adj	95.24	-	95.24	94.66	94.95
6	interp	99.98	-	99.96	99.62	99.79
7	punct	97.33	-	97.33	98.39	97.86
8	fin	97.68	-	97.68	98.88	98.27
9	ppron3	99.51	-	99.51	97.31	98.39
10	conj	92.36	-	92.36	91.42	91.89
11	ger	72.50	-	72.50	75.96	74.19
12	ppas	87.00	-	87.00	83.84	85.39
13	adv	91.46	-	91.46	95.52	93.44
14	impt	100	-	100	88.73	94.03
15	comp	90.69	-	90.69	96.04	93.29
16	inf	99.01	-	99.01	99.01	99.01
17	Bedzie	100	-	100	99.37	99.68
18	pred	61.47	-	58.03	78.88	66.86
19	:	100	-	100	100	100
20	brev	95.77	-	95.77	97.29	96.52
21	dot	86.30	-	86.30	84.25	85.26
22	num	90.04	-	81.30	87.29	84.18
23	pcon	99.08	-	99.08	97.29	98.17
24	ppron12	87.50	-	87.50	97.10	92.05
25	imps	100	-	100	100	100
26	pact	95.49	-	95.49	95.00	95.24
27	aglt	99.50	-	99.50	97.52	98.50
28	interj	92.31	-	92.31	51.85	66.40
29	numcol	100	-	100	98.88	99.43
30	siebie	100	-	100	99.27	99.63
31	adjc	100	-	100	100	100
32	winien	100	-	100	98.00	98.99
33	adja	93.34	-	93.34	93.33	93.33
34	adjp	92.86	-	92.86	86.47	89.55
35	depr	100	-	100	100	100
36	burk	100	50.00	72.50	62.5	67.13
37	xxx	100	13.21	57.02	66.7	61.48
38	-)	100	-	100	100	100
39	pant	100	-	100	100	100
Razem		95.05	52.94	92.47	92.23	92.97

Tabela 4: Dokładność rozpoznawania kategorii syntaktycznych z wykorzystaniem ukrytego modelu Markowa dedykowanego dla języka polskiego.
(Źródło: *opracowanie własne*).

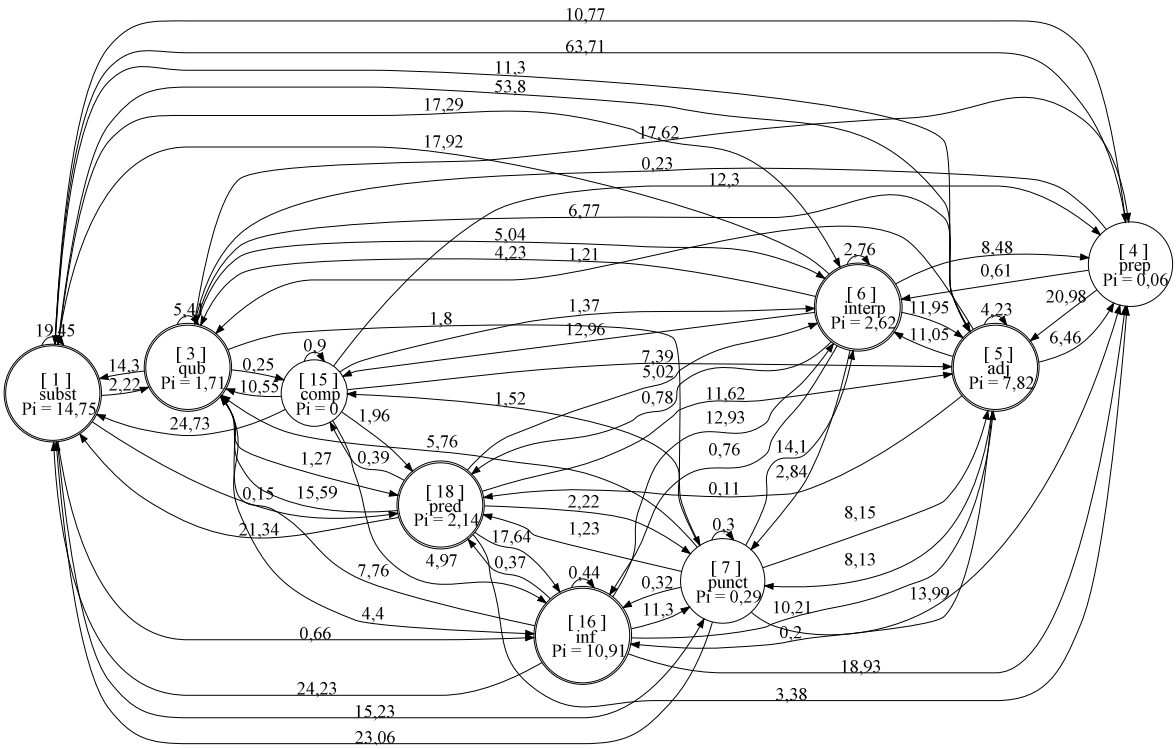
10.3.4 Wybrane przykłady rozpoznawania kategorii syntaktycznych dla języka polskiego

Ze względu na rozbudowana strukturę modelu Markowa służącego do rozpoznawania kategorii syntaktycznych, jego graficzna prezentacja (patrz Rysunek 5) ograniczona została tu jedynie do stanów wykorzystanych w przykładowym zdaniu. Niemniej jednak należy pamiętać, że całkowita przestrzeń stanów jest dużo większa. Przykład zdania w języku polskim:

„Trzeba niestety przyznać, że wielu ludzi nabiera się na takie kłamstwa.“



Rysunek 4: Przykład rozpoznawania kategorii syntaktycznych z wykorzystaniem ukrytego modelu Markowa.
(Źródło: *opracowanie własne*).



Rysunek 5: Model Markowa prezentujący prawdopodobieństwa przejść między wybranymi kategoriami syntaktycznymi.
(Źródło: *opracowanie własne*).

10.3.5 Model rozpoznawania kategorii zależnościowych dla języka polskiego

Reguły gramatyki języka polskiego różnią się od tych występujących w języku angielskim. Niemniej jednak pewne analogie w nazewnictwie i stosowanych zasadach pozostają ponadjęzykowe. Jedną z takich zasadach jest budowa zdania SVO (ang. *Subject Verb Object*) określająca kolejność wystąpienia podmiotu, orzeczenia i dopełnienia. Kategorie zależnościowe dla języka polskiego przypisane do stanów w ukrytym modelu Markowa przedstawia Tabela 5.

Numer stanu	Nazwa stanu	Opis	Część mowy	Przykłady językowe
0	unknown	Wyrażenie nieznane		
1	adjunct	Przydawka	prep conj adv adj qub subst num	*„koło“, „jak“, „jednak“, „lipca“, „już“, „100“*
2	comp	Uzupełnienie	prep num subst	*„tytułu“, „terenie“, „w“, „z“, „na moim biurku“, „do włosów“*
3	subj	Podmiot	adj num subst	*„jej nikt nie podskoczy“, „każdy z synów założył rodzinę“, „choć wszystko było“, „siedem płócien“*
4	app	Przyłożenie	subst	*„jego twarz, pan bóg“*
5	ne	Skrót	brev	*„r .“, “godz .“, „prof .“*
6	pred	Predykat	verb adj subst	*„robi groźne“, „został podany“, „ten pokój“*
7	obj	Dopełnienie bliższe	conj num subst	*„zmienia nazwisko“, „planów czy idei“, „które cię cieszą“, „możesz mieć więcej“*
8	punct	Znak koniec zdania	punct interp	*„ .“*
9	obj_th	Dopełnienie dalsze	subst adj	*„wydaje mi się“, „dać mu obietnicę“, „powiem wam że“*
10	comp_inf	Dopełniacz bezokolicznikowy	verb	*„da się zachować“, „może być“, „tylemówić o“*
11	pd	Predykatyw dopełniacza	adj verb subst	*„co było przyczyną“, „był wściekły“, „można założyć , że“*
12	refl	Znacznik refleksyjny	qub	*„zbiła się“, „zjawi się“*
13	conjunct	Koniunkcja koordynowana	adj prep num subst	*„mądry i uczynny“, „za dużo czy za mało“, „po miesiącu i trzech dniach“, „Bolka i Lolka“*
14	coord _punct	Koniunkcja interpunkcyjna	interp conj	*„przesadą, gdy“, „całej, dużej i“*
15	neg	Znacznik negacji	qub	*„nie zniosła“, „nie zdając“*
16	complm	Uzupełniacz	comp	*„, że ludzie“, „, iż żywe“*
17	comp_fin	Uzupełnienie okolicznikowe	conj adj verb	*„aby mogły wziąć“, „głośno i wyraźnie“, „gdyż każdego z“*
18	mwe	Wyrażenie wielowyrazowe	qub adj subst	*„to też wstyd“, „50 milionów euro“, „po prostu uznanie“*
19	aglt	Odmiana ruchoma	verb	*„jak mogł em“, „gdyby m wiedział“*
20	cond	Enklityka warunkowa	qub	*„stało by się“, „musiało by nastąpić“*
21	abbrev _punct	Znacznik skrótu	punct	*„art .“, „prof .“,*
22	coord	Koniunkcja koordynująca	conj qub	*„ale“, „a“, „również“, „jako“, „niż“*
23	aux	Pomocniczy	verb	*„może zostać“, „warto było“*

24	pre_coord	Pre-koniunkcja	conj	*„albo, to i tak“*
25	interp	Znak interpunkcyjny	interp	*„ ,“*
26	ne_		interp subst adj num brev	*„1992 r .“, „13 czerwca“, „np .“*
27	imp	Znacznik rozkazujący	qub, verb	*„niech pan“, „mów teraz“*

Tabela 5: Kategorie zależnościowe dla języka polskiego przypisane do stanów w ukrytym modelu Markowa.
(Źródło: *opracowane na podstawie (Wróblewska 2014)*).

10.3.6 Wyniki rozpoznawania kategorii zależnościowych dla języka polskiego

W wyniku uczenia określonych zostało 11 581 różnych obserwacji. Ocena modelu na zestawie testowym wykazała, że dla około 7.33% (tj. 785 spośród 10 702) wyrażeń nie udało się zidentyfikować pasującej obserwacji, stąd zostały one przypisane do kategorii „nieznana“. Precyzja rozpoznawania obserwacji nieznanych wyniosła 59.26%, natomiast dla znanych obserwacji precyzja wyniosła już 78.40%, tym samym całkowita precyzja klasyfikacji dla wszystkich obserwacji w zestawie testowym wyniosła 76.78%. Całkowita czułość klasyfikacji dla wszystkich obserwacji w zestawie testowym wyniosła 78.54%. Stąd, miara F1 dla przedstawionego klasyfikatora wyniosła 77.35%.

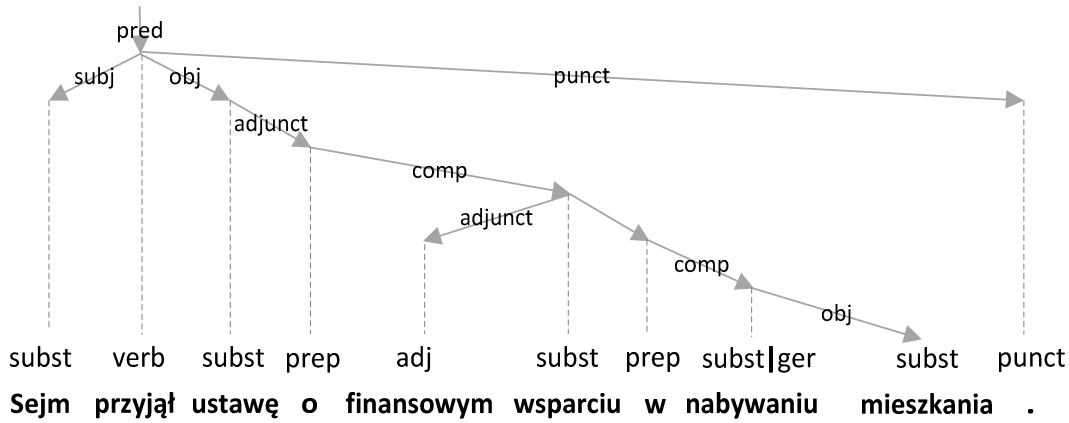
		Warstwowy ukryty model Markowa, dla którego obserwacje stanowią kombinacje afiksów wyrazów i odkrytych dla nich kategorii syntaktycznych				
Numer stanu	Nazwa stanu	Precyzja dla znanych [%]	Precyzja dla nieznanych [%]	Precyzja wszystkich [%]	Czułość wszystkich [%]	Miara F1 wszystkich [%]
0	unknown	-	-	-	-	-
1	adjunct	81.02	75.49	79.86	83.25	81.52
2	comp	72.41	57.26	68.34	70.17	69.24
3	subj	80.08	67.11	72.29	65.72	68.85
4	app	73.70	-	73.70	73.88	73.79
5	ne	74.63	-	74.63	74.25	74.44
6	pred	71.73	59.81	62.11	80.25	70.02
7	obj	72.46	47.86	66.26	68.51	67.36
8	punct	91.74	-	91.74	98.94	95.20
9	obj_th	83.77	-	83.77	79.06	81.34
10	comp_inf	87.60	81.34	86.16	85.03	85.59
11	pd	81.83	-	81.83	82.90	82.36
12	refl	95.53	-	95.53	100	97.71
13	conjunct	81.30	25.92	67.84	61.77	64.66
14	coord_punct	41.50	-	41.50	44.09	42.75
15	neg	95.59	-	95.59	100	97.74
16	complm	81.67	-	81.67	88.87	85.11
17	comp_fin	74.34	-	74.34	86.67	80.03
18	mwe	84.26	-	84.26	67.92	75.21
19	aglt	95.63	-	95.63	100	97.77
20	cond	98.00	-	98.00	100	98.99
21	abbrev_punct	66.19	-	66.19	100	79.65
22	coord	20.37	-	20.37	19.23	19.78
23	aux	80.28	-	80.28	59.74	68.50
24	pre_coord	50.17	-	50.17	50.00	50.08
25	interp	100	-	100	100	100
26	ne_	100	-	100	100	100
27	imp	81.02	-	81.02	80.50	80.75
Razem		78.40	59.26	76.78	78.54	77.35

Tabela 6: Dokładność rozpoznawania kategorii zależnościowych z wykorzystaniem ukrytego modelu Markowa dedykowanego dla języka polskiego.
(Źródło: *opracowanie własne*).

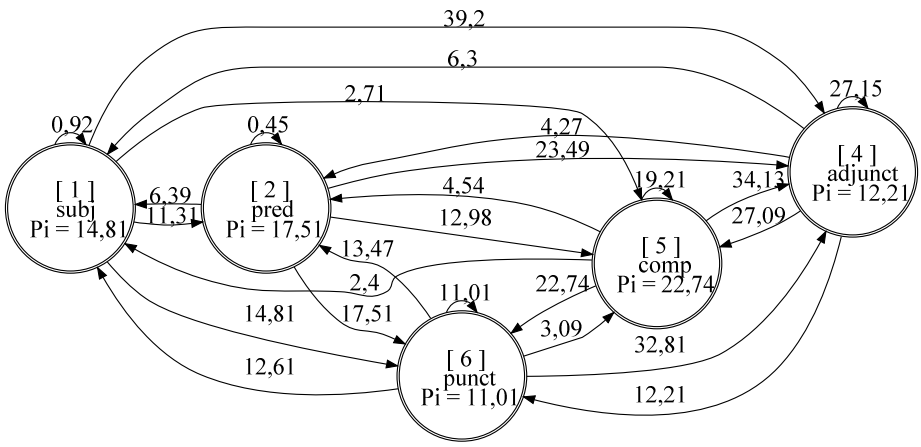
10.3.7 Wybrane przykłady rozpoznawania kategorii zależnościowych dla języka polskiego

Ze względu na rozbudowana strukturę modelu Markowa służącego do rozpoznawania kategorii zależnościowych , jego graficzna prezentacja (patrz Rysunek 7 i 9) ograniczona została tu jedynie do stanów wykorzystanych w przykładowym zdaniu. Niemniej jednak należy pamiętać, że całkowita przestrzeń stanów jest dużo większa. Przykład zdania w języku polskim:

„Sejm przyjął ustawę o finansowym wsparciu w nabywaniu mieszkania.“



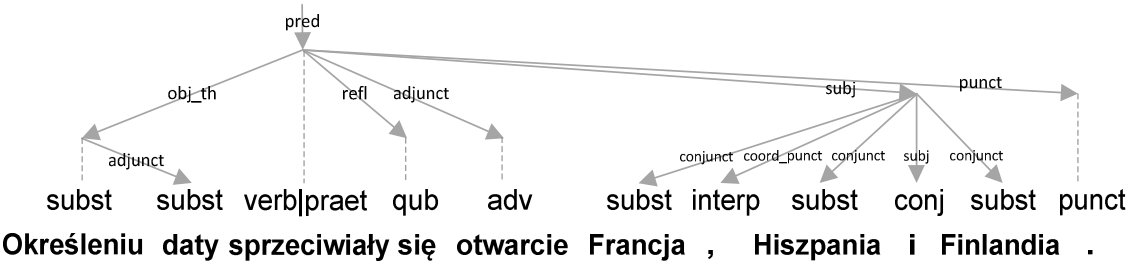
Rysunek 6: Przykład rozpoznawania części mowy i części zdania w wykorzystaniem ukrytych modeli Markowa.
(Źródło: *opracowanie własne*).



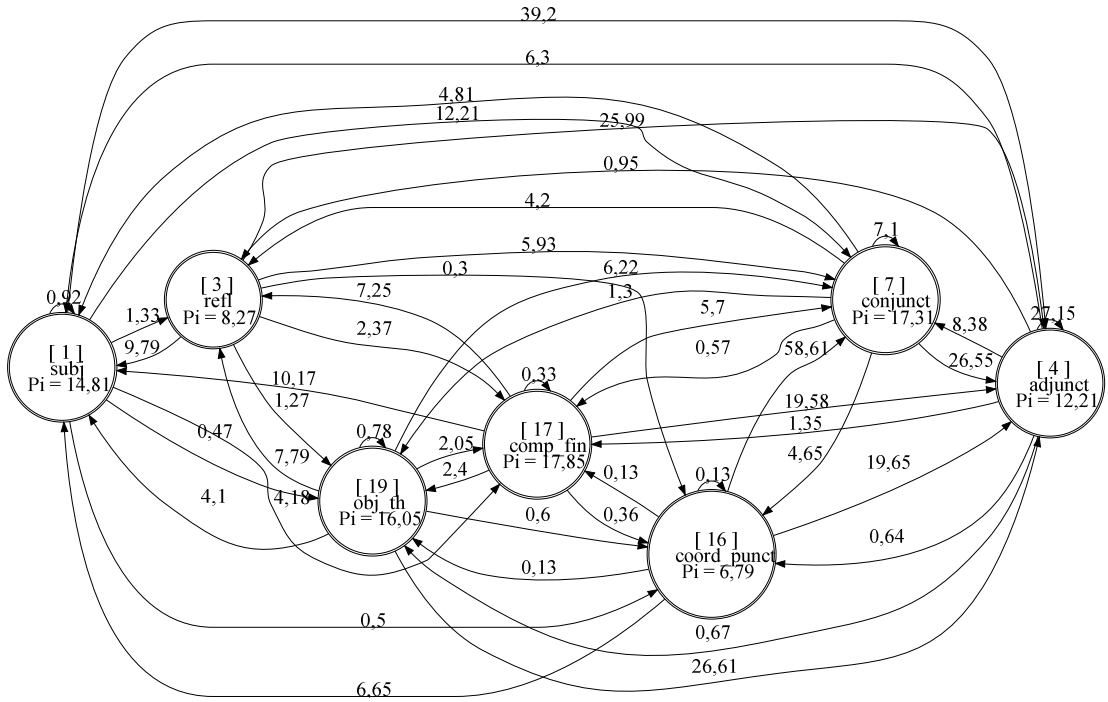
Rysunek 7: Model Markowa prezentujący prawdopodobieństwa przejść miedzy wybranymi kategoriami zależnościowymi.
(Źródło: *opracowanie własne*).

Przykład zdania w języku polskim:

„Określeniu daty sprzeciwiały się otwarcie Francja, Hiszpania i Finlandia.“



Rysunek 8: Przykład rozpoznawania kategorii zależnościowych z wykorzystaniem warstwowych ukrytych modeli Markowa.
(Źródło: *opracowanie własne*).



Rysunek 9: Model Markowa prezentujący prawdopodobieństwa przejść miedzy wybranymi kategoriami zależnościowymi.
(Źródło: *opracowanie własne*).

10.4 Bezpieczeństwo numeryczne ukrytych modeli Markowa o zredukowanej reprezentacji liczbowej

Jak już wspomniano w (Rabiner 1989), obliczenia w ramach HMM mogą stać się kłopotliwe, zwłaszcza w przypadku długich sekwencji. Nawet jeśli obliczenia te są wykonywane w przestrzeni logarytmicznej, należy pamiętać, że istnieje zawsze pewna granica długości badanej sekwencji, poza którą obliczenia mogą stać się numerycznie niestabilne. Zjawisko to należy rozpatrywać w parze z rozmiarem (szerokością bitową) reprezentacji liczb zmiennoprzecinkowych. W celu implementacji FPGA w dalszej części pracy zastosowana zostanie redukcja standardowej reprezentacji o podwójnej precyzji (64 bity) do precyzji: pojedynczej (32 bity), połówkowej (16 bitów) lub ćwiartkowej (8 bitów). Przedstawiony poniżej lemat stanowi główny teoretyczny wynik pracy. Pokazuje on ilościowo, w jaki sposób można ustalić długość najkrótszej sekwencji niebezpiecznej numerycznie w terminach skrajnych prawdopodobieństw występujących w modelu HMM oraz przy pomocy zadanej szerokości bitowej mantysy dla reprezentacji zmiennoprzecinkowej. Wynik ten został opublikowany w pracy (Pietras & Klęsk 2017).

10.4.1 Lemat „o długości sekwencji niebezpiecznej numerycznie“

Niech $p > 0$ i $P > 0$ oznaczają odpowiednio najmniejsze niezerowe i największe prawdopodobieństwa występujące w modelu HMM (tj. skrajne prawdopodobieństwa z macierzy przejść, emisji lub rozkładu początkowego). Załóżmy, że obliczenia są wykonywane w ramach logarytmowanej wersji algorytmu Viterbiego, a model jest przechowywany w ograniczonej reprezentacji zmiennoprzecinkowej z wykorzystaniem mantysy o szerokości m. Wtedy, najkrótsza sekwencja niebezpieczna numerycznie – tj. taka, która może powodować błędy numeryczne wynikające z ograniczonej reprezentacji nie zaś standardowe błędy zaokrągleń – jest długości

$$\Theta\left(2^{m-(\log_2|\log p|-\log_2|\log P|)+1}\right). \tag{1}$$

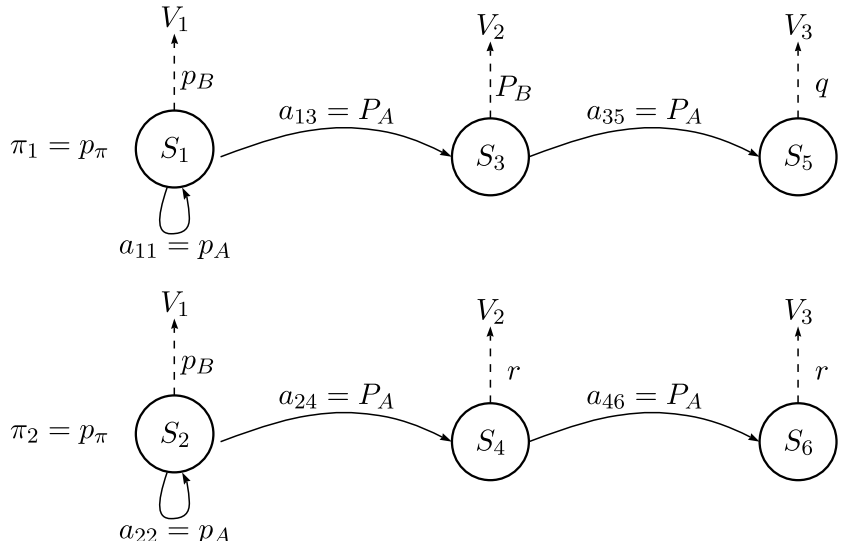
Dla przypomnienia warto podkreślić, że chodzi tu o dokładny rząd asymptotyczny w odróżnieniu od notacji dużego O (górne ograniczenie asymptotyczne) czy też dużego Ω (dolne ograniczenie asymptotyczne). Innymi słowy, jeżeli w naszym przypadku mówimy, że niebezpieczna sekwencja ma długość $T^* = \Theta\left(f(m,p,P)\right)$, gdzie f to funkcja w powyższym wzorze, to oznacza to, że istnieją pewne dwie stałe c_1, c_2 takie że okładna wartość T^* mieści się w widełkach: $c_1 \cdot f(m,p,P) \leqslant T^* \leqslant c_2 \cdot f(m,p,P)$.

Szkic dowodu: Zaaranżujemy sytuację, w której dla pewnej sekwencji obserwacji możliwe będą tylko dwie ścieżki Viterbiego o bardzo zbliżonych prawdopodobieństwach. Początkowy długi fragment sekwencji będzie wskazywał na remis pomiędzy ścieżkami, dopiero dwie ostatnie obserwacje przeważą szalę na rzecz jednej ze ścieżek (w przypadku obliczeń dokładnych). Pokażemy, że w przypadku obliczeń numerycznych w ramach logarytmowanej wersji algorytmu Viterbiego możliwe będzie rozstrzygnięcie przeciwne.

Dowód: W notacji małe litery będą wskazywały na małe prawdopodobieństwa (i skojarzone z nimi logarytmy), duże litery będą wskazywały na duże prawdopodobieństwa (i ich logarytmy). Rozważmy minimalne i maksymalne prawdopodobieństwa w macierzach π, A, B (uwaga: w przypadku minimalnych, chodzi o minimalne niezerowe):

$$\begin{aligned}
&p_\pi = \min\left\{\pi_i : 1 \leqslant i \leqslant N, \pi_i > 0\right\}, && l_\pi = \log p_\pi,\\
&p_A = \min\left\{a_{ij} : 1 \leqslant i,j \leqslant N, a_{ij} > 0\right\}, && l_A = \log p_A,\\
&p_B = \min\left\{b_{ik} : 1 \leqslant i \leqslant N,\ 1 \leqslant k \leqslant M, b_{ik} > 0\right\}, && l_B = \log p_B,\\
&P_A = \max\left\{a_{ij} : 1 \leqslant i,j \leqslant N\right\}, && L_A = \log P_A,\\
&P_B = \max\left\{b_{ik} : 1 \leqslant i \leqslant N,\ 1 \leqslant k \leqslant M\right\}, && L_B = \log P_B.
\end{aligned} \tag{2}$$

Rozważmy szczególny model HMM, w którym ważne są tylko stany i emisje przedstawione przez Rysunek 10.



Rysunek 10: Szczególny model HMM dla lematu.
(Źródło: *opracowanie własne*).

Należy zaznaczyć, że przejścia pomiędzy stanami S_1 i S_2 są niemożliwe, $a_{12} = a_{21} = 0$; a także że wszystkie inne stany modelu $S_{i>6}$ (pominięte na rysunku) mają zerowe prawdopodobieństwa emisji dla obserwacji: V_1, V_2, V_3. Oprócz przejść wskazanych przez Rysunek 10 stany $S_1, ..., S_6$ mogą wykonywać przejścia do innych stanów $S_{i>6}$, jednakże te nie będą ważne w przykładzie. Prawdopodobieństwa emisji q, r można traktować jak parametry, które zostaną odpowiednio dobrane dla rozstrzygnięcia remisu. Niech $l_q = \log q$ i $l_r = \log r$. Przypuśćmy, że na wejściu pojawiła się następująca sekwencja obserwacji do zbadania:

$$\underbrace{(V_1, V_1, \ldots, V_1}_{T-2}, V_2, V_3).$$

Łatwo zauważyć, że możliwe są tylko dwie ścieżki stanów, które mogły wyprodukować powyższą sekwencję. Są to:

$$(S_1, S_1, ..., S_1, S_3, S_5)\,,$$
$$(S_2, S_2, ..., S_2, S_4, S_6)\,.$$

Nazwijmy je odpowiednio ścieżką „nieparzystą“ i „parzystą“ (ze względu na indeksy biorące w nich udział). Iloczyn prawdopodobieństw i odpowiadająca mu suma logarytmów dla ścieżki „nieparzystej“ wynoszą:

$$\underbrace{p_\pi \cdot p_B}_{t=1} \cdot \underbrace{p_A \cdot p_B}_{t=2} \cdot \ldots \cdot \underbrace{p_A \cdot p_B}_{t=T-2} \cdot \underbrace{P_A \cdot P_B}_{t=T-1} \cdot \underbrace{P_A \cdot q}_{t=T}, \tag{3}$$

$$\underbrace{l_\pi + l_B}_{t=1} + \underbrace{l_A + l_B}_{t=2} + \ldots + \underbrace{l_A + l_B}_{t=T-2} + \underbrace{L_A + L_B}_{t=T-1} + \underbrace{L_A + l_q}_{t=T}. \tag{4}$$

Analogicznie dla ścieżki „parzystej“ mamy:

$$\underbrace{p_\pi \cdot p_B}_{t=1} \cdot \underbrace{p_A \cdot p_B}_{t=2} \cdot \ldots \cdot \underbrace{p_A \cdot p_B}_{t=T-2} \cdot \underbrace{P_A \cdot r}_{t=T-1} \cdot \underbrace{P_A \cdot r}_{t=T}, \tag{5}$$

$$\underbrace{l_\pi + l_B}_{t=1} + \underbrace{l_A + l_B}_{t=2} + \ldots + \underbrace{l_A + l_B}_{t=T-2} + \underbrace{L_A + l_r}_{t=T-1} + \underbrace{L_A + l_r}_{t=T}. \tag{6}$$

Przypuśćmy, że w rzeczywistości bardziej prawdopodobna jest ścieżka „parzysta“ (tzn. w przypadku obliczeń dokładnych). Oznacza to, że spełniona jest następująca nierówność (po uproszczeniu wyrazów remisowych i zredukowaniu do samej końcówki):

$$P_B \cdot q < r^2. \tag{7}$$

Jednocześnie chcielibyśmy pokazać możliwość błędu numerycznego polegającego na zdarzeniu, że algorytm wskaże jako lepszą ścieżkę „nieparzystą“. Błąd taki może wystąpić wtedy, gdy w chwili $t = T-1$ dodanie wyrazu L_B (bliskiego zeru ponieważ P_B jest maksymalne) do bieżącej sumy, która jest odpowiednio dużego rzędu, nie zmieni wyniku. A zatem chcemy spowodować przeciwny kierunek nierówności w (7) zastępując równoważnie P_B jedynką ze względu na nierozróżnialność wyniku, tj.:

$$1 \cdot q > r^2, \tag{8}$$

co, patrząc na sumę logarytmów, odpowiada:

$$\log 1 + \log q > 2 \log r. \tag{9}$$

Zatem, aby spełnić jednocześnie (7) i (8) należy dobrać q jako:

$$r^2 < q < r^2/P_B. \tag{10}$$

Dla uproszczenia rozważań numerycznych zaniedbajmy teraz rozróżnienie pomiędzy prawdopodobieństwami pochodzącymi z A, B lub π; tj. utożsamijmy $p_A = p_B = p_\pi = p$ oraz $P_A = P_B = P_\pi = P$. Tym samym niech $l_A = l_B = l_\pi = l$ oraz $L_A = L_B = L_\pi = L$. Innymi słowy interesują nas tylko małe i duże wyrazy (co do rzędu wielkości) w sumie logarytmów. I tak suma (4) dla ścieżki „nieparzystej“ upraszcza się do

$$\underbrace{l+l}_{t=1} + \underbrace{l+l}_{t=2} + \ldots + \underbrace{l+l}_{t=T-2} + \underbrace{L+L}_{t=T-1} + \underbrace{L+l_q}_{t=T}, \tag{11}$$

a suma (6) dla ścieżki „parzystej“ do

$$\underbrace{l+l}_{t=1} + \underbrace{l+l}_{t=2} + \ldots + \underbrace{l+l}_{t=T-2} + \underbrace{L+l_r}_{t=T-1} + \underbrace{L+l_r}_{t=T}. \tag{12}$$

Przed krytycznym momentem $t = T-1$ bieżąca wartość obu powyższych sum powstaje poprzez dodanie $2(T-2)$ logarytmów z małych prawdopodobieństw, czyli wartości bezwzględnie dużych; łatwo wymusić $|l| \gg |L|$. A zatem staramy się w szybkim tempie „wyczerpać“ m cyfr znaczących (w bazie dwójkowej) mantysy tuż przed dodaniem składnika L, który jest małego rzędu. Wielkość $\log_2 |L|$ można traktować właśnie jako *rząd wielkości* L w bazie dwójkowej. A zatem wspomnianą sytuację błędu, gdy następuje brak rozróżnienia wyniku po dodaniu L, można opisać nierównością:

$$\frac{2^{m-|\log_2|L||+1}}{2\,(T-2)\,|l|} \leqslant 1. \tag{13}$$

Licznik ułamka reprezentuje rząd wielkości maksymalnej możliwej do zapisania liczby na mantysie szerokości m po uwzględnieniu odjęcia $|\log_2|L||$ cyfr znaczących krytycznego składnika. Mianownik reprezentuje tempo, w jakim wyczerpujemy ten dostępny rząd wielkości, poprzez $2(T-2)$-krotne dodawanie l w początkowej fazie obliczeń. Rozwiązując (13) ze względu na T (dlogość sekwencji) otrzymujemy:

$$\begin{aligned} T &\geqslant 2 + \tfrac{2^{m-|\log_2|L||+1}}{2|l|} \\ &= 2 + 2^{m+\log_2|L|+1} \cdot 2^{\log_2|2l|^{-1}} \\ &= 2 + 2^{m+\log_2|L|-\log_2|2l|+1} \\ &= 2 + 2^{m-(\log_2|2l|-\log_2|L|)+1} = T^*. \end{aligned} \tag{14}$$

Póki co pokazane zostało, że mogą istnieć sekwencje niebezpieczne numerycznie o długości większej od lub równej pewnej wartości progowej, tj. przypadek : $T \geqslant T^*$. Biorąc pod uwagę sens notacji dużego Θ w lemacie, aby dokończyć dowód, należy jeszcze pokazać przypadek $T \leqslant T^*$– ograniczając tym samym z góry najkrótszą spośród takich sekwencji. Tu wystarczy zauważyć, że jeżeli w naszym przykładzie w początkowym fragmencie ścieżki weźmiemy pod uwagę prawdopodobieństwa większego rzędu niż minimalne p, to dodawane logarytmy prawdopodobieństw nie wyczerpią tak szybko dostępnego zakresu. Tym samym sytuacja błędu nierozróżnialnego wyniku może co najwyżej wystąpić później. A zatem *najkrótsza* niebezpieczna sekwencja jest długości $\Theta(T^*)$. ■

Uwaga (1).
Ze wzoru na T^* (14) widać, że ważna jest różnica pomiędzy rzędami wielkości logartymów (tj. $\log_2 |2l| - \log_2 |L|$) ze skrajnych prawdopodobieństw w modelu HMM. Im ta różnica jest większa, tym mniej pozostaje swobodnych cyfr znaczących w mantysie.

Uwaga (2).
O ile zewnętrzny logarytm oddający rząd wielkości musi być dwójkowy – $\log_2 \ldots$ – tak wewnętrzne logarytmy schowane w $l = \log p$ i $L = \log P$ mogą być wyrażone w dowolnej podstawie.

10.4.2 Wpływ zredukowania precyzji zmiennoprzecinkowej modeli na długość badanej sekwencji

Dla wszystkich przedstawionych ukrytych modeli Markowa określona została, względem zredukowanych precyzji zmiennoprzecinkowych, maksymalna długość badanej sekwencji do której zagwarantowane jest bezpieczeństwo numeryczne obliczeń. Zestawienie wpływu zastosowania lematu dla modeli HMM dedykowanych dla syntaktycznej i semantycznej analizy danych tekstowych (wyliczanych w połówkowej precyzji obliczeniowej) przedstawia Tabela 7.

Nazwa ukrytego modelu Markowa	Bezpieczna długość badanej sekwencji	Procent błędów rozpoznawania bez uwzględnienia lematu	Procent błędów rozpoznawania z uwzględnieniem lematu
Model z obserwacją agregowaną binarnie dla ekstrakcji z HTML	1 048	5.35% (824/15 400)	0.26% (41/15 400)
Model z obserwacją agregowaną wzrażeniowo dla ekstrakcji z HTML	121	7.79% (1 201/15 400)	0.49% (76/15 400)
Model rozpoznawania części mowy dla j. angielskiego	213	18.70% (8 859/47 370)	0.26% (124/47 370)
Model rozpoznawania grup frazowych dla j. angielskiego	925	21.50% (10 194/47 400)	0.25% (120/47 400)
Model rozpoznawania części zdania dla j. angielskiego	32	77.40% (44 648/57 680)	0.44% (256/57 680)
Model rozpoznawania części mowy dla j. polskiego	332	16.71% (12 396/74 182)	0.39% (295/74 182)
Model rozpoznawania części zdania dla j. polskiego	37	21.35% (2 285/10 702)	0.64% (69/10 702)

Tabela 7: Zestawienie wpływu zastosowania lematu dla modeli HMM prezentowanych w rozprawie wyliczanych w połówkowej precyzji obliczeniowej.
(Źródło: *opracowanie własne*).

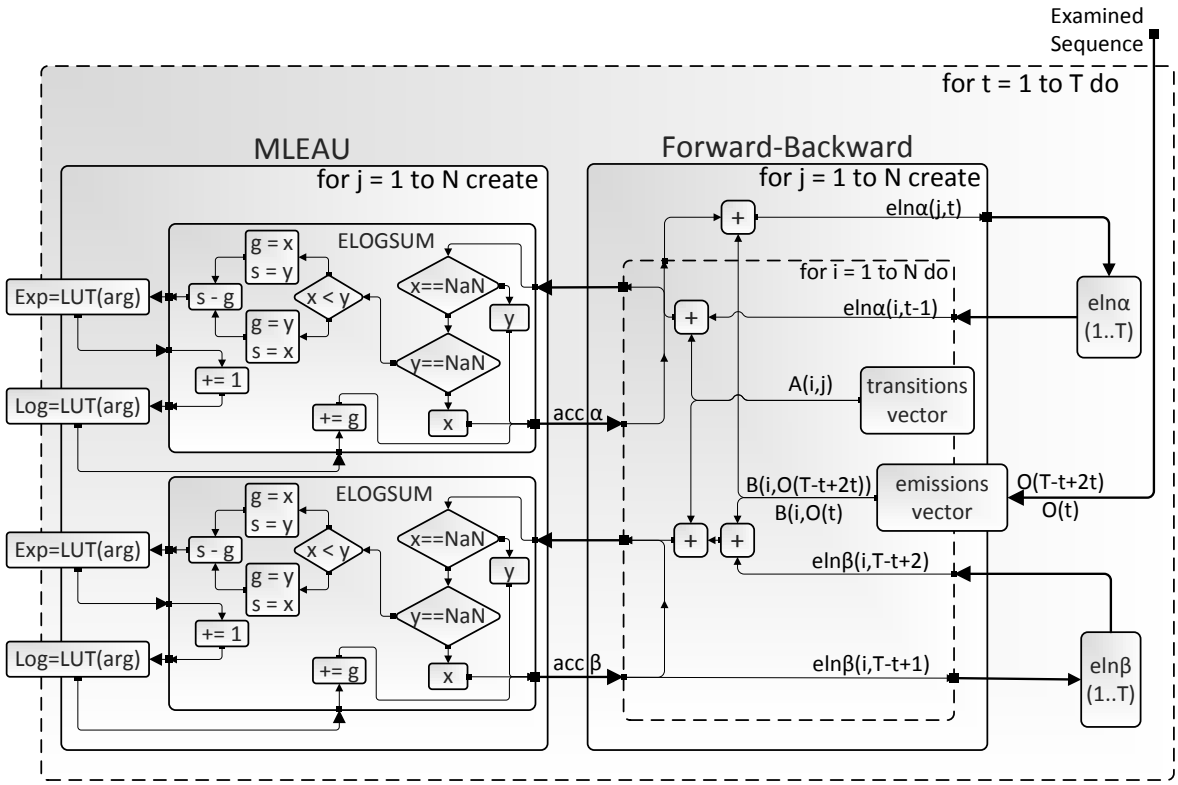
Nie uwzględniając lematu, rozpoznawanie zostało wykonane na pełnej długości sekwencji obserwacji (np. o długości 15 400). Natomiast przy uwzględnieniu lematu długość badanej sekwencji obserwacji była nie dłuższa niż bezpieczna długość sekwencji wskazane przez zastosowanie lematu dla danego modelu. Pomimo zastosowania lematu nadal w obliczeniach o zredukowanej precyzji może dochodzić do błędów zaokrągleń, niemniej jednak wyniki przedstawione w Tabeli 7 dowodzą, że zastosowanie lematu jest konieczne dla zmniejszenia liczby błędnie rozpoznanych kategorii.

10.5 Sprzętowa realizacja algorytmów uczenia maszynowego

10.5.1 Blok Forward-Backward D&C

Realizacja algorytmu Forward-Backward w dziedzinie logarytmów nie wymaga współczynników skalowania (znanych z tradycyjnej wersji tego algorytmu), ponieważ w dziedzinie logarytmów obliczenia są bardziej stabilne numerycznie i nie ma potrzeby stosowania dodatkowych ulepszeń. Warto zauważyć, że bez współczynników skalowania obliczanie do przodu i od tyłu mogą być wykonywane równocześnie. Z punktu widzenia sprzętowej implementacji algorytmu współczynniki skalowania wymagają nie tylko dodatkowego nakładu obliczeń, ale także muszą być przechowywane w pamięci FPGA dla całej badanej sekwencji obserwacji do dalszych obliczeń algorytmu Bauma–Welcha.

Rysunek 11) przedstawiona diagram wizualizacji bloku algorytmicznego Forward-Backward D&C. Zarówno t jak i kolejne wizualizacje bloków algorytmicznych w spójny sposób przedstawiają najważniejsze cechy proponowanej struktury obliczeniowej.

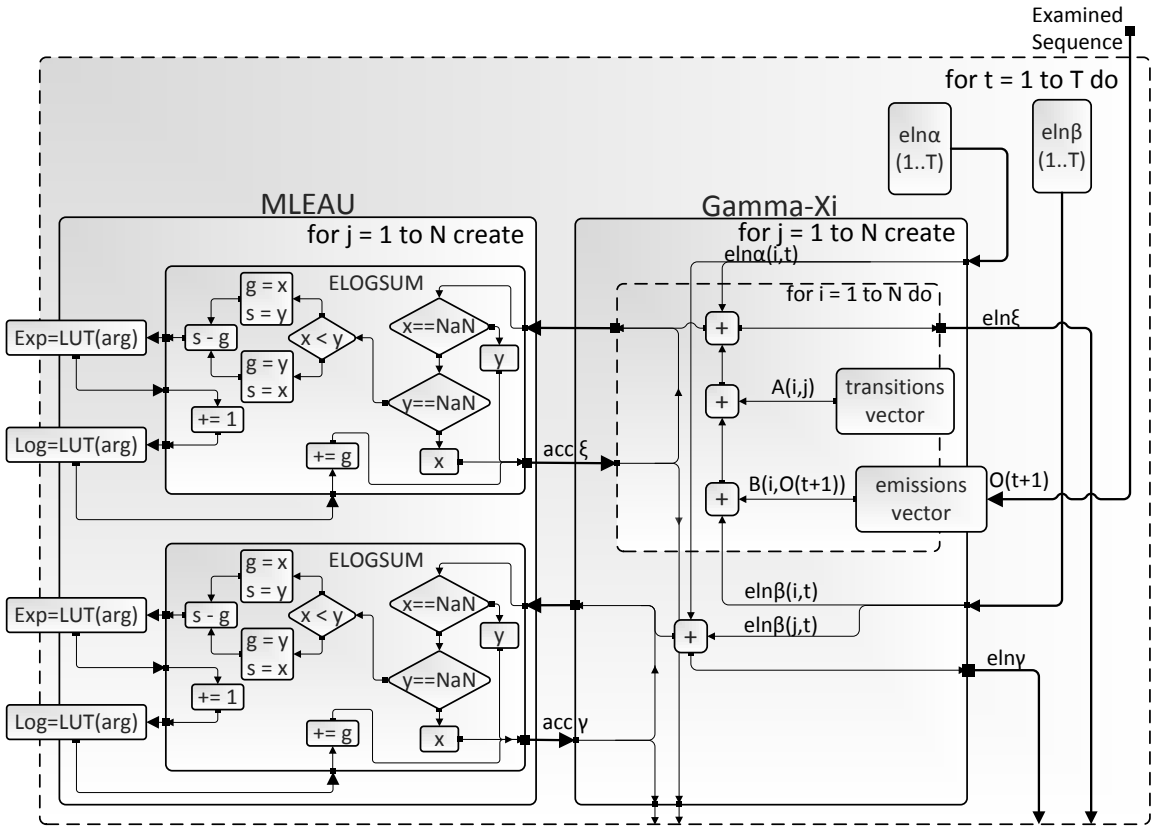


Rysunek 11: Wizualizacja bloku algorytmicznego przeznaczonego pod obliczenia związane z algorytmem Forward-Backward D&C.
(Źródło: *opracowanie własne*).

Blok algorytmiczny produkuje wartości α i β dla każdego stanu względem badanej sekwencji obserwacji. Wektory α i β są przechowywane w FPGA do dalszych obliczeń przeprowadzanych przez blok algorytmiczny Gamma-Xi.

10.5.2 Blok Gamma-Xi D&C

Wartości γ oraz ξ odgrywają kluczową rolę podczas szacowania parametrów modelu HMM. W sprzętowym wariancie dokonana została reorganizacja pracy algorytmu Bauma–Welcha taka, że reestymacja nowych parametrów HMM następować będzie krokowo. Zamiast wyliczać γ i ξ dla całej badanej sekwencji obserwacji, wystarczające jest określenie ich wartości cząstkowych, co przedstawia Rysunek 12.

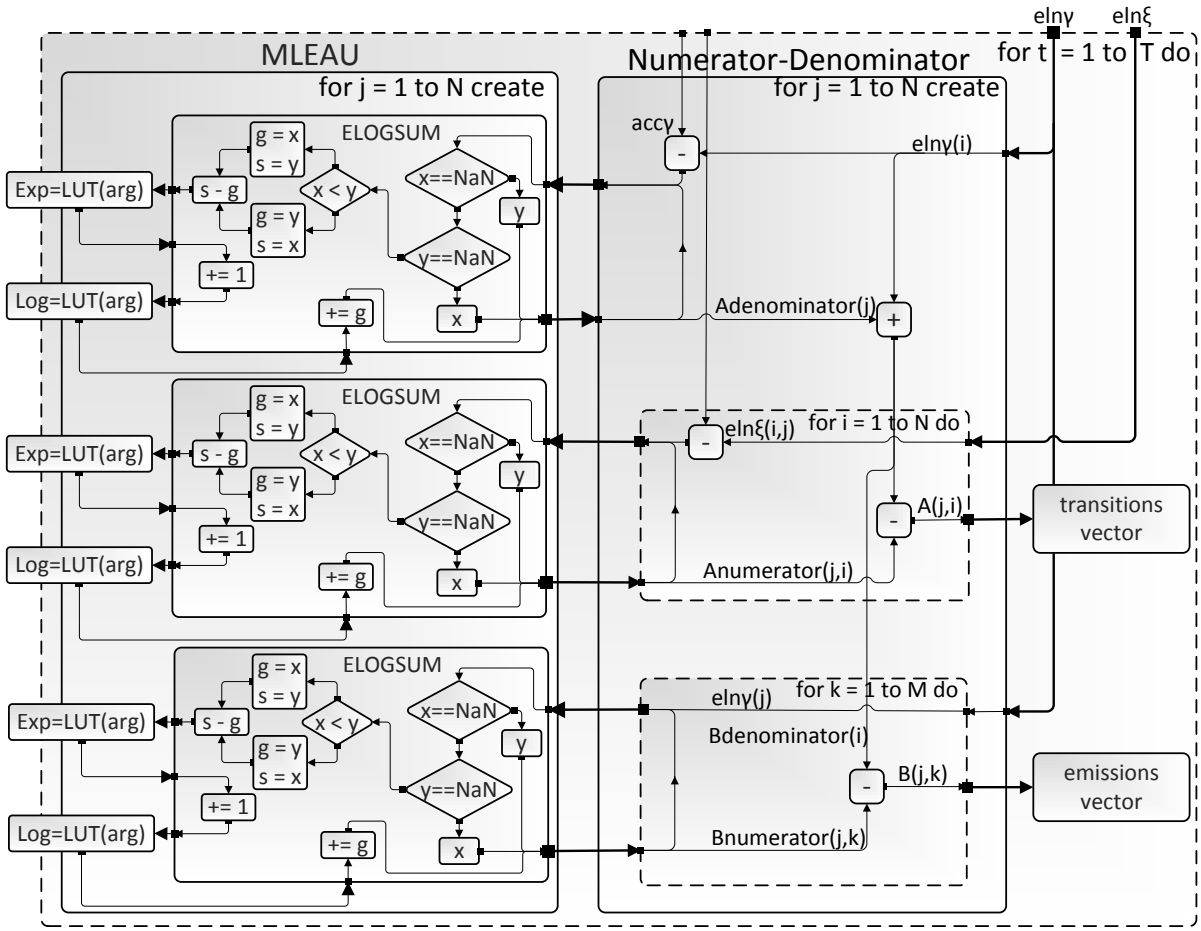


Rysunek 12: Wizualizacja bloku algorytmicznego przeznaczonego pod obliczenia związane z algorytmem Gamma-Xi D&C.
(Źródło: *opracowanie własne*).

W każdej kolejnej chwili czasowej t obliczane i normalizowane są jedynie cząstkowe wartości γ i ξ względem aktualnie badanej obserwacji o_t i na ich podstawie, dokonywane jest również cząstkowa wyliczenie licznika i mianownika w bloku Numerator-Denominator. Jednakże cząstkowe wartości licznika i mianownika muszą być przechowywane i kumulowane oddzielnie dla każdego stanu i dla każdej obserwacji. Oznacza to, że pamięć FPGA musi być w stanie jednocześnie przechowywać zarówno bieżące parametry HMM (A, B, π), jak i nowe (A', B', π'). Końcowa reestymacja parametrów odbywa się w bloku algorytmicznym Numerator-Denominator.

10.5.3 Blok Numerator-Denominator D&C

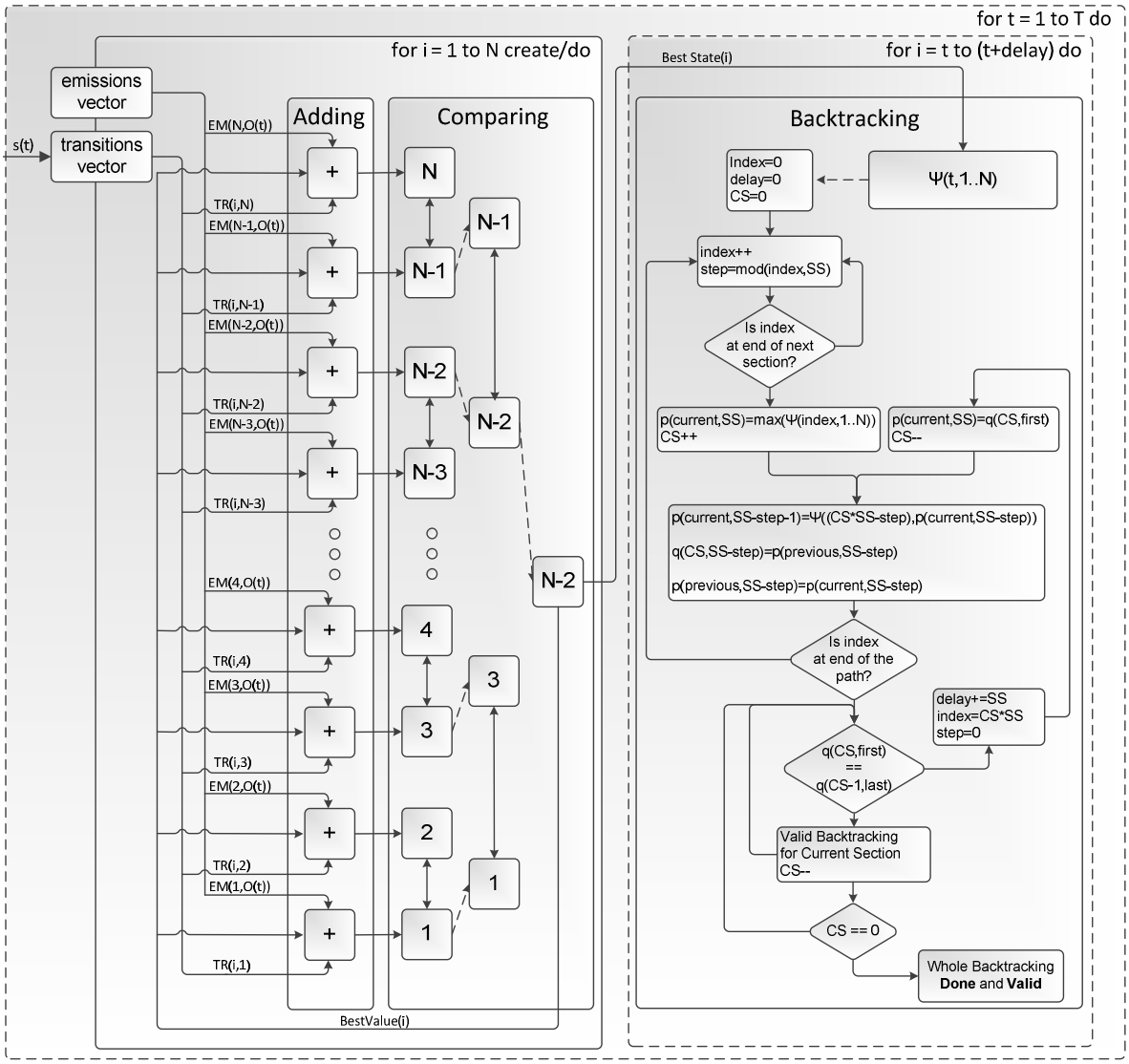
Obliczenia bloku algorytmicznego Numerator-Denominator są zorganizowane podobnie jak w bloku Gamma-Xi. Cząstkowe wartości γ i ξ są akumulowane odpowiednio w liczniku i mianowniku oddzielnie dla każdego stanu i dla każdej obserwacji (patrz Rysunek 13). Oznacza to, że pamięć FPGA musi być w stanie jednocześnie przechowywać zarówno bieżące parametry HMM (A, B, π), jak i nowe (A', B', π'). Po pełnym zakumulowaniu wartości cząstkowych (po osiągnięciu ostatniej obserwacji w badanej sekwencji) dokonywana jest ostateczna normalizacja.



Rysunek 13: Wizualizacja bloku algorytmicznego przeznaczonego pod obliczenia związane z algorytmem Numerator-Denominator D&C.
(Źródło: *opracowanie własne*).

10.5.4 Sprzętowa realizacja algorytmu Viterbi D&C

W celu uniknięcia problemów numerycznych algorytm Viterbiego również jest obliczany w dziedzinie logarytmów, w której mnożenie prawdopodobieństw przejść, emisji i wartości maksymalnej zastępowane jest operacją sumowania. Sumowanie odbywa się równolegle dla wszystkich stanów. Ostatnim etapem jest znalezienie stanu o najwyższej wartości prawdopodobieństwa przejścia. W tym celu wystarczające jest porównanie wszystkich prawdopodobieństw przejscia, jednakże ta prosta procedura wymaga wykonania n (liczba stanów) porównań sekwencyjnie. Przyspieszenie tego kroku realizowane jest, przez zastosowanie metody: dziel i zwyciężaj. Ogólna idea jest przedstawiona na Rysunek 14, gdzie wyróżnione są trzy bloki, a linie przerywane przedstawiają iteracyjne wykorzystanie zasobów. W bloku sumacji „Adding“, wykonywane jest potrójne sumowanie zmiennoprzecinkowa dla wartości maksymalnej (uzyskanej z poprzedniej iteracji) oraz prawdopodobieństwa emisji i prawdopodobieństw przejścia. W bloku porównywania „Comparing“, wszystkie stany są porównywane równolegle parami, co pozwala na wykonanie procedury w $\log_2 n + 1$ kroków (gdzie n odnosi się do liczby stanów). Podejście tutaj jest analogiczne do algorytmu sortowania merge-sort (Dhawan & Dehon 2015), który wymaga $O(n \log n)$ cykli obliczeniowych, aby uporządkować wszystkie wartości w n-elementowym wektorze. W przypadku algorytmu Viterbiego ważne jest znalezienie wartości maksymalnej wektora, stąd tylko $O(\log_2 n)$ cykli obliczeniowych jest koniecznych.

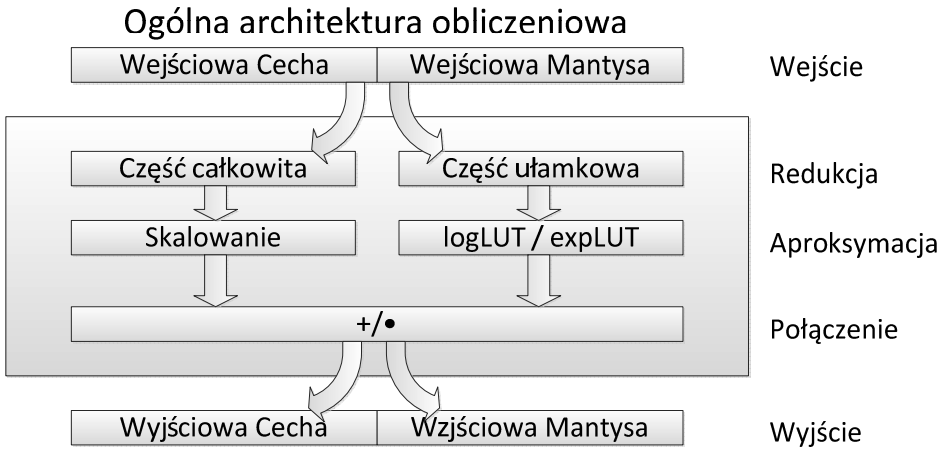


Rysunek 14: Wizualizacja bloku algorytmicznego przeznaczonego pod obliczenia związane z algorytmem Viterbi D&C.
(Źródło: *opracowanie własne*).

Technika D&C stosowana jest również przy wykonywaniu bloku algorytmu nawracania w celu zredukowania opóźnienia związanego z identyfikację prawidłowej ścieżki stanów dla całej długości badanej sekwencji obserwacji. W tym przypadku nawracanie dla długich sekwencji podzielone zostanie na równe kawałki. W ten sposób procedura nawracania jest wykonywana na kolejnych odcinkach równolegle z wyliczaniem kolejnej iteracji (w przód) w algorytmie Viterbiego.

10.5.5 MLEAU — jednostka aproksymacji wielu funkcji logarytmicznych i wykładniczych

W celu skutecznego (z wysoką wydajnością i zadowalającą precyzją obliczeń) przeprowadzenia kosztownych obliczeniowo operacji w dziedzinie logarytmów, utworzona została specjalna jednostka aproksymacji wielu funkcji logarytmicznych i wykładniczych MLEAU (ang. *Multiple Logarithm Exponent Approximation Unit*). Dla przypomnienia — rozszerzona arytmetyka logarytmiczna odbywa się według funkcji określonych w pracy (Mann 2006). Dla sprawnego przeprowadzenia tego typu obliczeń proponowana jest jednostka MLEAU stanowiąca rozszerzenie algorytmu ICSIlog (Vinyals & Friedland 2008) przez dodanie funkcjonalność przekształcenia wykładniczego oraz interfejs równoległego dostępu, co przekłada się na zysk w wydajności.



Rysunek 15: Ogólna architektura bloku aproksymacji dla przekształcenia funkcji logarytmicznej i wykładniczej.
(Źródło: *opracowanie własne*).

Przekształcenie logarytmiczne i wykładnicze jako operacje odwrotne wymagają podobnych działań. W związku z tym ich szybka aproksymacja opiera się na operacji indeksowania tablicy, w którym mantysa argumentu zmiennoprzecinkowego przekształcana jest za pomocą tablicy przeglądowej (LUT) do odpowiedniej wartości.

11 Podsumowanie

W rozprawie przedstawiona została sprzętowa implementacja algorytmów programowania dynamicznego stosowanych w ukrytych modelach Markowa przeznaczonych do syntaktycznej i semantycznej analizy danych tekstowych. Na potrzeby prac badawczych opracowana została aplikacja HMM-Toolbox, pozwalająca na tworzenie, zarządzanie, uczenie i testowanie ukrytych modeli Markowa.

Dla pozyskiwania istotnych danych tekstowych ze źródeł HTML zaprojektowano model Markowa, proponując dwa style obserwacji z przestrzeni międzytekstowej: znaczniki grupowane binarnie i znaczniki grupowane w wyrażenia. Nauczanie modeli przeprowadzono etapami w sposób nadzorowany z uwzględnieniem kroku związanego z treningiem Viterbiego. Etap znakowania treści witryn internetowych przeprowadzono z wykorzystaniem funkcji zbiorczego oznaczania treści w aplikacji HMM-Toolbox. Oba modele wykazały bardzo wysoką dokładność rozpoznawania treści właściwej (powyżej 98%), przy czym obserwacja oparta o grupowanie binarne lepiej radziła sobie przy klasyfikacji nieznanych przestrzeni międzytekstowych. Cecha ta jest o tyle ważna, że przy ocenie zmieniającej się w czasie struktury witryny internetowej nauczony model przez dłuższy okres dostarcza poprawnych wyników klasyfikacji. Opracowana metoda pozyskiwania istotnych danych ze źródeł HTML bazuje głównie na przestrzeni międzytekstowej (zestawy znaczników HTML), a tym samym jest niezależna od języka witryny internetowej.

W zadaniach oznaczania tekstów częściami mowy i częściami zdania opracowano wiele autorskich modeli Markowa różniących się zarówno przestrzenią stanów jak i zestawem możliwych obserwacji. Niemniej jednak dla wszystkich modeli zaproponowano jednolity styl obserwacji na podstawie morfemów brzegowych tworzonych przez zadanej długości afiksy wyrazów z możliwością stopniowego wygaszania wcześniejszych afiksów. Ten rodzaj obserwacji pozwala na określenie precyzji odwzorowania wyrazów, co pozwala na zredukowanie liczności dla n-gramowego zestawu obserwacji.

Przy oznaczaniu tekstu częściami mowy obserwacje w ukrytym modelu Markowa stanowiły morfemy brzegowe będące nośnikiem informacji o znaczeniu syntaktycznym wyrazu. Nauczanie modeli wykonane zostało w oparciu o bank drzew składniowych (dla języka angielskiego Penn Treebank, a dla języka polskiego Składnica frazowa). Dla obydwu języków osiągnięto wysoką dokładność rzędu 95% dla wyrazów grupowanych w ramach znanych afiksów.

Dla języka angielskiego dokonany został podział zdania na części stanowiące grupy frazowe. Dokonano oceny skuteczności rozpoznawania grup frazowych przez pojedynczy jak i dwuwarstwowy ukryty model Markowa. W pojedynczym modelu przestrzeń stanów i obserwacji bezpośrednio wynika z bazy danych uczących. Dla dwuwarstwowego ukrytego modelu Markowa, przestrzeń obserwacji na wyższym poziomie zawiera informacje zarówno z afiksów wyrazów, jak i informacje o części mowy odkrytej przez model Markowa z warstwy niższej. Dokładność klasyfikacji czterech najbardziej znaczących grup frazowych z wykorzystaniem dwuwarstwowego modelu wyniosła około 93%, podczas gdy stosując pojedynczy model, otrzymywany wynik rozpoznawania był o 10 punktów procentowych słabszy.

Warstwowe ukryte modele Markowa zostały zastosowane również przy rozpoznawaniu typów relacji zależnościowych w zdaniu. Zarówno w języku angielskim jak i polskim model Markowa w niższej warstwie dostarcza informacji o rozpoznanej części mowy, natomiast model z wyższej warstwy dokonuje końcowej klasyfikacji części zdania. Zaproponowana warstwowa struktura klasyfikacji osiągnęła zadowalający wynik, pozwalając na prawidłowe rozpoznanie co najmniej siedmiu z dziesięciu zadanych kategorii zależnościowych.

Informacje o znaczeniu syntaktycznym wyrazów w zdaniu uzyskane w wyniku klasyfikacji części mowy (przy pomocy ukrytego modelu Markowa) pozwoliły na disambiguację syntaktyczną, co jest niezbędne przy korzystaniu z sieci semantycznej typu Wordnet. Umożliwiło to trafne odkrywanie hiponimii i hiperonimii wyrazowej, a tym samym opis informacji na różnych poziomach semantyki leksykalnej. Informacje o funkcji wyrazów w zdaniu dostarczone w wyniku przypisania im kategorii zależnościowych (przy pomocy warstwowych ukrytych modeli Markowa)

pozwoliły na odkrywanie relacji semantycznych niezbędnych przy tworzeniu opisu semantycznego.

Przedstawione algorytmy programowania dynamicznego wykorzystywane w obliczeniach ukrytych modeli Markowa zostały z powodzeniem zrealizowane sprzętowo w logice programowalnej układu FPGA. Struktura algorytmiczna została utworzona wykorzystując syntezę na wysokim poziomie HLS, natomiast kosztowne obliczeniowo przekształcenia logarytmiczne i wykładnicze realizowane są według autorskiego pomysłu przez wyspecjalizowaną autorską jednostkę aproksymacji MLEAU utworzoną w języku opisu sprzętu. Tym samym powstała architektura zapewnia wysoką wydajność obliczeniową przy jednoczesnym zachowaniu elastyczności. Dzięki HLS możliwe jest łatwe dopasowanie architektury do ilości i rodzaju alokowanej pamięci, wykorzystywanej precyzji obliczeniowej oraz stopnia zrównoleglenia.

Przeprowadzanie efektywnych i bezpiecznych numerycznie obliczeń algorytmów dedykowanych dla ukrytych modeli Markowa o zredukowanej reprezentacji zmiennoprzecinkowej wymaga nałożenia pewnych ograniczeń na długość badanej sekwencji obserwacji. Przedstawiony i udowodniony w rozprawie lemat o bezpieczeństwie numerycznym ukrytych modeli Markowa nadaje ramy obliczeniowe gwarantujące stabilność numeryczną i jednocześnie stanowi uzasadnienie stawianej w rozprawie tezy.

12 Spis publikacji własnych

1. Pietras, M. (2013), Knowledge extraction and decision rules generation, based on rough set theory in order to increase the direct marketing campaigns effectiveness, in P. Jałowiecki, E. Jałowiecka, P. Łukasiewicz i A. Orłowski, eds, 'Information Systems in Management XVIII', WULS Press, Warsaw, Poland.

2. Pietras, M. (2014a), Hardware conversion of neural networks simulation models for neural processing accelerator implemented as fpga-based soc, in '2014 24th International Conference on Field Programmable Logic and Applications (FPL)', pp. 1–4.

3. Pietras, M. (2014b), 'Sentence sentiment classification using fuzzy word matching combined with fuzzy sentiment classifier', Electrical Review - Special issue Vol. 91(2), pp. 107–111.

4. Pietras, M. (2015), Error analysis in the hardware neural networks applications using reduced floating-point numbers representation, in 'Conference: AIP Conference Proceedings', Vol. 1648, p. 660005.

5. Pietras, M. (2017), Hidden Markov Models with affix based observation in the field of syntactic analysis, in J. Kacprzyk, I. E. Fray, J. Pejaś, A. Piegat, S. ya Kobayashi, J. Kacprzyk, I. E. Fray, J. Pejaś, A. Piegat i S. ya Kobayashi, eds, 'Hard and Soft Computing for Artificial Intelligence, Multimedia and Security', Vol. 534, Springer, Cham, pp. 17–26.

6. Pietras, M. i Klęsk, P. (2017), 'FPGA implementation of the logarithmic versions of baum-welch and viterbi algorithms for reduced precision Hidden Markov Models', Bulletin of the Polish Academy of Sciences: Technical Sciences Vol.65(6), 935–947.

Spis rysunków

Spis tabel

Literatura

Abney, S. & Light, M. (1999), Hiding a Semantic Class Hierarchy in a Markov Model, in 'Proceedings of ACL Workshop on Unsupervised Learning in NLP. University of Maryland'.

Aithal, S. (2016), 'A survey on partial reconfiguration techniques', International Journal Of Engineering And Computer Science **5**(5), 16613–16616.

Baroni, M. & Zamparelli, R. (2010), Nouns are vectors, adjectives are matrices: Representing adjective-noun constructions in semantic space, in 'Proceedings of the 2010 Conference on Empirical Methods in Natural Language Processing', Cambridge, Massachusetts, pp. 1183–1193.

Bos, J. (2011), 'A survey of computational semantics: Representation, inference and knowledge in wide-coverage text understanding', Language and Linguistics Compass **5/6**, 336–366.

Cambria, E. & White, B. (2014), 'Jumping nlp curves: A review of natural language processing research', IEEE Computational Intelligence Magazine **9**, 48–57.

Cohn, T. & Blunsom, P. (2005), Semantic role labelling with tree conditional random fields, in 'Proceedings of the Ninth Conference on Computational Natural Language Learning', Ann Arbor, Michigan, pp. 169–172.

Dai, Q., Chen, E. & Shi, L. (2009), An iterative approach for joint dependency parsing and semantic role labeling, in 'Proceedings of the Thirteenth Conference on Computational Natural Language Learning: Shared Task', Boulder, Colorado, pp. 19–24.

Dhawan, U. & Dehon, A. (2015), 'Area-Efficient Near-Associative Memories on FPGAs', ACM Transactions on Reconfigurable Technology and Systems (TRETS) **7**, 30–41.

Eisner, J. M. (1996), Three new probabilistic models for dependency parsing: An exploration, in 'Proceedings of the 16th conference on Computational linguistics', Copenhagen, Denmark, pp. 340–345.

Ellson, J., Gansner, E., Koutsofios, L., North, S., Woodhull, G., Description, S. & Technologies, L. (2001), Graphviz — open source graph drawing tools, in 'Lecture Notes in Computer Science', Springer-Verlag, pp. 483–484.

Fillmore, C. J. (1968), The case for case, in E. Bach & R. T. Harms, eds, 'Universals in Linguistic Theory', Holt, Rinehart and Winston, London.

Gao, C. & Xu, B. (2013), 'The application of semantic field theory to english vocabulary learning', Theory and Practice in Language Studies **3**(11), 2030–2035.

Ge, R. & Mooney, R. J. (2005), A statistical semantic parser that integrates syntax and semantics, in 'Proceedings of the Ninth Conference on Computational Natural Language Learning', Ann Arbor, Michigan, pp. 9–16.

Gesmundo, A., Henderson, J., Merlo, P. & Titov, I. (2009), A latent variable model of synchronous syntactic-semantic parsing for multiple languages, in 'Proceedings of the Thirteenth Conference on Computational Natural Language Learning: Shared Task', Boulder, Colorado, pp. 37–42.

Grefenstette, E., Blunsom, P., de Freitas, N. & Hermann, K. M. (2014), 'A deep architecture for semantic parsing', arXiv preprint arXiv:1404.7296 .

Gupta, S., Agrawal, A., Gopalakrishnan, K. & Narayanan, P. (2015a), Deep Learning with Limited Numerical Precision, in 'Proceeding ICML'15 Proceedings of the 32nd International Conference on International Conference on Machine Learning', Lille, France, pp. 1737–1746.

Gupta, S., Agrawal, A., Gopalakrishnan, K. & Narayanan, P. (2015b), Deep learning with limited numerical precision, in 'ICML'.

Hajic, J., Ciaramita, M., Johansson, R. & et al. (2009), The CoNLL-2009 shared task: Syntactic and semantic dependencies in multiple languages, in 'Proceedings of the Thirteenth Conference on Computational Natural Language Learning: Shared Task', Boulder, Colorado, pp. 1–18.

Hajnicz, E. (2014), Lexico-Semantic Annotation of Składnica Treebank by means of PLWN Lexical Units, in 'Proceedings of the 7th International WordNet Conference (GWC 2014)', Tartu, Estonia, pp. 23–31.

Henderson, J., Merlo, P., Musillo, G. & Titov, I. (2008), A latent variable model of synchronous parsing for syntactic and semantic dependencies, in 'Proceedings of the Twelfth Conference on Computational Natural Language Learning', Manchester, United Kingdom, pp. 178–182.

Henderson, J., Merlo, P., Titov, I. & Musillo, G. (2013), 'Multi-lingual joint parsing of syntactic and semantic dependencies with a latent variable model', Computational Linguistics **39**, 949–998.

Hulden, M. (2012), Treba: Efficient numerically stable em for pfa, in 'ICGI'.

Iwakura, T. & Okamoto, S. (2008), 'A fast boosting-based learner for feature-rich tagging and chunking', pp. 17–24.

Lecun, Y., Bengio, Y. & Hinton, G. (2015), 'Deep learning', Nature **521**, 436–444.

Leonard E. Baum, Ted Petrie, G. S. & Weiss, N. (1970), 'A maximization technique occurring in the statistical analysis of probabilistic functions of markov chains', The Annals of Mathematical Statistics **41**(1), 164–171.

Lluis, X. (2008), Joint Learning of Syntactic and Semantic Dependencies, PhD thesis, Universitat Politecnica de Catalunya, Barcelona.

Lluis, X., Bott, S. & Marquez, L. (2009), A second-order joint Eisner model for syntactic and semantic dependency parsing, in 'Proceedings of the Thirteenth Conference on Computational Natural Language Learning: Shared Task', Boulder, Colorado, pp. 79–84.

Lluis, X. & Marquez, L. (2008), A joint model for parsing syntactic and semantic dependencies, in 'Proceedings of the Twelfth Conference on Computational Natural Language Learning', Manchester„ pp. 188–192.

Llus, X., Carreras, X. & Marquez, L. (2013), 'Joint arc-factored parsing of syntactic and semantic dependencies', Transactions of the Association for Computational Linguistics **1**, 219–230.

Mann, T. P. (2006), 'Numerically stable hidden markov model implementation. An HMM scaling tutorial'.

Maziarz, M., Piasecki, M., Rudnicka, E., Szpakowicz, S. & Kędzia, P. (2016), plWordNet 3.0 – a Comprehensive Lexical-Semantic Resource, in 'Proceedings of COLING 2016', Osaka, Japan, pp. 2259–2269.

McCallum, A., Freitag, D. & Pereira, F. C. N. (2000), Maximum entropy Markov models for information extraction and segmentation, in 'Proceeding ICML '00 Proceedings of the Seventeenth International Conference on Machine Learning', pp. 591–598.

Meyers Adam, R. & Macleod, C. (2008), 'Nombank v 1.0 ldc2008t23'.

Meza-Ruiz, I. & Riedel, S. (2009), Jointly identifying predicates, arguments and senses using Markov logic, in 'Proceedings of Human Language Technologies: The 2009 Annual Conference of the North American Chapter of the Association for Computational Linguistics', Boulder, Colorado, pp. 155–163.

Ovtcharov, K., Ruwase, O., Kim, J.-Y., Fowers, J., Strauss, K. & Chung, E. (2015), 'Accelerating Deep Convolutional Neural Networks Using Specialized Hardware'.

Pado, S. & Lapata, M. (2009), 'Cross-lingual annotation projection for semantic roles', Journal of Artificial Intelligence Research **36**, 307–340.

Palmer, M., Gildea, D. & P., P. K. (2006), 'The proposition bank: An annotated corpus of semantic roles', Computational linguistics **31**, 71–106.

Pietras, M. (2014), Hardware conversion of neural networks simulation models for neural processing accelerator implemented as fpga-based soc, in '2014 24th International Conference on Field Programmable Logic and Applications (FPL)', pp. 1–4.

Pietras, M. (2015), Error analysis in the hardware neural networks applications using reduced floating-point numbers representation, in 'Conference: AIP Conference Proceedings', Vol. 1648, p. 660005.

Pietras, M. & Klęsk, P. (2017), 'Fpga implementation of the logarithmic versions of baum-welch and viterbi algorithms for reduced precision hidden markov models', Bulletin of the Polish Academy of Sciences: Technical Sciences **65**(6), 935–947.

Przepiórkowski, A., Bańko, M., Górski, R. L. & Lewandowska-Tomaszczyk, B. (2012), in 'Narodowy Korpus Języka Polskiego', Wydawnictwo Naukowe PWN.

Punyakanok, V., Roth, D. & t. Yih, W. (2008), 'The importance of syntactic parsing and inference in semantic role labeling', Computational Linguistics **34**, 257–287.

Rabiner, L. R. (1989), A tutorial on hidden Markov models and selected applications in speech recognition, in 'Proceedings of the IEEE', pp. 257–286.

Samuelsson, Y., Tackstrom, O., Velupillai, S. & et al. (2008), Mixing and blending syntactic and semantic dependencies, in 'Proceedings of the Twelfth Conference on Computational Natural Language Learning', Manchester, United Kingdom, pp. 248–252.

Schmidhuber, J. (2015), 'Deep Learning in Neural Networks: An Overview', Neural Networks **61**, 85–117.

Sun, W., Li, H. & Sui, Z. (2008), The integration of dependency relation classification and semantic role labeling using bilayer maximum entropy Markov models, in 'Proceedings of the Twelfth Conference on Computational Natural Language Learning', Manchester, United Kingdom, pp. 243–247.

Surdeanu, M., Johansson, R., Meyers, A. & et al. (2008), The CoNLL-2008 shared task on joint parsing of syntactic and semantic dependencies, in 'Proceeding CoNLL '08 Proceedings of the Twelfth Conference on Computational Natural Language Learning', Manchester, United Kingdom, pp. 159–177.

Taylor, A., Marcus, M. & Santorini, B. (2003), 'The penn treebank: An overview'.

Titov, I. & Henderson, J. (2007), A latent variable model for generative dependency parsing, in 'Proceedings of the 10th International Conference on Parsing Technologies', Prague, Czech Republic, pp. 144–155.

Vinyals, O. & Friedland, G. (2008), 'A Hardware-independent fast logarithm approximation with adjustable accuracy', Multimedia, ISM 2008, Tenth IEEE International Symposium on pp. 61–65.

Wróblewska, A. (2014), Polish Dependency Parser Trained on an Automatically Induced Dependency Bank, PhD thesis, Institute of Computer Science, Polish Academy of Sciences, Warsaw, Poland.

Wróblewska, A. (2018), 'Results of the poleval 2018 competition: Dependency parsing shared task', Ogrodniczuk M. and Kobyliński Ł. (eds.) Proceedings of the PolEval 2018 Workshop. Institute of Computer Science, Polish Academy of Sciences **7**, 11–24.

Wróblewska, A. & Przepiórkowski, A. (2014), Projection-based annotation of a polish dependency treebank, in 'Proceedings of the Ninth International Conference on Language Resources and Evaluation (LREC'14)', Reykjavik, Iceland, pp. 26–31.

Yunfei, G. (2017), Investigating Read/Write Aggregation to Exploit Power Reduction Opportunities Using Dual Supply Voltages, Master's thesis, School of Engineering and Applied Science, Washington, USA.

Zhang, C., Li, P., Sun, G., Guan, Y., Xiao, B. & Cong, J. (2015), Optimizing FPGA-based Accelerator Design for Deep Convolutional Neural Networks, in 'Proceeding FPGA '15 Proceedings of the 2015 ACM/SIGDA International Symposium on Field-Programmable Gate Arrays', Monterey, California, USA, pp. 161–170.