Detekce přízvuků v ruštině s použitím klasifikátoru

Anastasiia Chizhova [1]

Úvod

V současné době se naše katedra zabývá zpracováním řeči, jejím rozpoznáváním a syntézou. Při převodu textu do syntetizované řeči se setkáme s obrovským počtem komplikací, které přímo souvisí s daným jazykem, který právě používáme. Já se zabývám ruštinou, konkrétně slovy, které mají nejednoznačný přízvuk, tzv. homografy. Přízvuk takových slov je závislý na kontextu a je ho třeba různými způsoby syntetizovat. Jako příklad uveďme 2 různé věty se stejnými slovy «году»:

- «в следующем году́» (*čes. V příštím roce)*
- «к 2010 - му го́ду» (*čes. Do roku 2010*).

Pro většinu slov lze používat slovník, ale pro homografy existuje několik případů, protože umístění přízvuku ovlivňuje význam slova. Cílem mého experimentu je predikovat přízvuk pouze na základě textu.

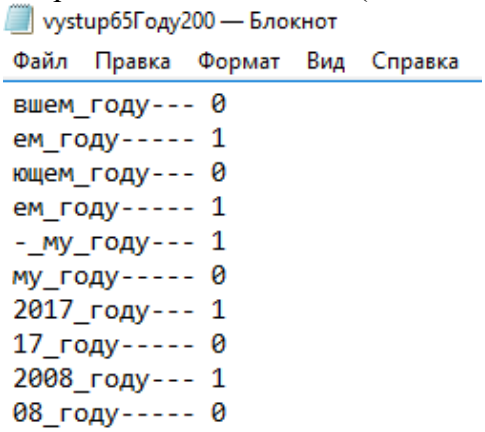
Při zpracování daného problému používám metodu “Učení s učitelem” (Psutka (2016)).

Příprava trénovacích dat

Z velkého počtu novinových textů byl sestaven list nejčastěji používaných slov, z nichž jsem vybrala několik nejednoznačných a pro každé z nich našla 100 vět s různými případy kontextu a přízvuku. Ve větách jsem ručně označila v daném slově přízvuk pomocí apostrofu před přízvučnou samohláskou. Dále jsem to musela klasifikovat do dvou tříd:

- je přízvuk (1)
- není přízvuk (0)

Nejprve jsem pro každou samohlásku daného slova vytvořila řetězec s L předchozími a P následujícími znaky a k němu přiřadila třídu 0 nebo 1 (viz. Obr. 1).



Obrázek 1 Výstupní soubor s označením klasifikace pro velikost kontextu zleva L-6 a zprava P-5

[1] studentka bakalářského studijního programu Inženýrská informatika, obor Inteligentní komunikace člověk - stroj, e-mail: chizhova@students.zcu.cz

Klasifikátory a klasifikace

Řetězce znaků jsem převedla na číselné vektory pomocí metody DictVectorizer z balíčku scikit-learn (Pedregosa *et al.* (2011)). Každá položka vektoru tak odpovídala výskytu konkrétního znaku na konkrétní pozici v řetězci. Zkoušela jsem 2 klasifikátory: Logistic Regression (LogReg) a Support Vector Machine (SVM), kde pro SVM jsem zkoušela 2 různá nastavení: rbf a linear. Proto, abych získala co nejpřesněji úspěšnost klasifikátoru, použila jsem křížovou validaci pomocí Leave-One-Out, což funguje tak, že ze všech dat vždy vynechá jednu položku na testovaní a ze zbytku se klasifikátor natrénuje, přičemž se použijí všechny možné kombinace trénovacích a testovacích dat. Testovala jsem různě dlouhé levé a pravé kontexty. Výsledky klasifikace jsem porovnala se správnými odpověďmi pomocí funkce f1_score (viz. Tab. 1).

Typ klasif.	LogReg			SVM		
Kontext	L-20 P-5	L-10 P-10	L-5 P-8	L-20 P-5	L-10 P-10	L-5 P-8
Году	86,57%	86,57%	84,73%	88,89%	89,00%	80,38%
Слова	88,00%	87,44%	87,50%	82,83%	85,02%	87,62%
Стороны	86,73%	88,66%	83,72%	86,70%	89,23%	82,57%

Tabulka 1 Výsledky klasifikátoru LogReg a SVM(linear) (f1_score)

Závěr

Z výsledků experimentu je zřejmé, že je možné pouze z textového okolí slova s nejednoznačným přízvukem s relativně vysokou úspěšností určit pozici přízvuků ve slově. Lepším z používaných klasifikátoru byl LogisticRegression, který pro všechna testovaná slova udává úspěšnost od 63,03% až do 88,66%. Maximální úspěšnost klasifikátoru SVM(rbf) je 80,73% a SVM(linear) je 89,23%. A pro většinu slov platí, že důležitější je levý kontext. Dále jsem už zkusila zvětšit počet trénovacích dat, což ve většině případů zvýší úspěšnost klasifikátoru, a mám v plánu přepočítat úspěšnost klasifikátoru pro celá slova.

Poděkování

Příspěvek byl podpořen projektem Ministerstva školství, mládeže a tělovýchovy číslo LO1506.

Literatura

Pedregosa, F., Varoquaux, G., Gramfort, A., Michel, V., Thirion, B., Grisel, O., Blondel, M., Prettenhofer, P., Weiss, R., Dubourg, V., Vanderplas, J., Passos, A., Cournapeau, D., Brucher, M., Perrot, M.,a Duchesnay, E. (2011) Scikit-learn: Machine Learning in Python, Journal of Machine Learning Research, 12, pp. 2825-2830.

Psutka, J. (2016) Učební texty z předmětu Základy strojového učeni a rozpoznávaní, ZČU v Plzni.