An Evolutionary Computing Approach Toward the Lemmatization of Textual Data

Amber Gillenwaters, Rafail Islam, and Jesse Simpson

The goal of our CSC 742 course project is to explore the application of evolutionary computing on the lemmatization of textual data in conjunction with machine learning in order to test and assess potential opportunities for optimization. Though common in practice, when working with large datasets, repetitive manual adjustment is inefficient and unlikely to result in optimal results. Fine-tuning hyperparameters on various machine learning algorithms in textual data is particularly challenging, as the theoretical bases for natural language processing are still under debate and not fully developed. The application of evolutionary programming toward the language classification problem domain in order to optimize machine learning algorithms is a suitable target. As this project is exploratory, the eventual experimental design will consider and may incorporate multiple parameters associated with word classification and word discrimination. Schmidt et al. (2019) demonstrates the capacity of differential evolution in hypertuning parameters allowing for a multimodal design. That said, focus on tuning a single procedural step may be preferable, allowing for more trials and greater detail.

This project has the potential for supporting or challenging the linear discrimination approach proposed by Baayen, Chuang, Shafaei-Bajestan, & Blevins (2019). For instance, Baayen et al. acknowledge that the application of a naïve Bayes classification model or neural network comparison may serve to supplement their findings. Alternatively, Lee, Lim, & Ahn (2019) suggest that graph-directed models may improve on tree-based models. Gleim et al. (2019) assessed lemmatization comparing fine-grained parts of speech as opposed to coarser groups, a parameter with potential for optimization using genetic algorithms. Baayen et al. and Buchanan used the tree-tagger library developed by Schmid (1994, 1999). Though tree-tagger is common in research, the library is deserving of review as it is more well-suited to smaller datasets. Adjusting decision trees by applying genetic algorithms using a random forest model may increase performance or contribute to improvements in word classification, producing stop-words, spell-checking, parts of speech, and semantic meaning most closely approximating natural language processing.

The data used are from Buchanan et al. (2019) consisting of over 16,000 concept-feature responses. In a concept-feature task, respondents are provided with a cue consisting of a single word to which they are asked to provide a typed response. These responses are analyzed and nominally classified. Words were classified as part of speech, including adjectives, nouns, and verbs. These were further classified as either concrete or abstract terms. Concrete terms are more easily visualized and sensed, while abstract terms require more higher-order thinking. The process identifies stop words, generates lemmas, groups of words with properties in common. Relationship encoding is based on distance. Because the data has been processed and labeled, the set is well-suited for experimentation.

The population will consist of lemmas produced, with variation in hyperparameters and value ranges within the decision tree or alternative classification method. The dependent variables may include computational speed, cost, and either alignment or improvement upon preexisting models (such as a reduction in dimensionality). In order to implement this project, sci-kit learn (Pedregosa et al*.*, 2011), NumPy (Oliphant, 2006), and pandas (McKinney, 2010) will be the primary existing packages utilized, with variation in logical application. These libraries are open-source and well-established in literature. In order to represent the problem in code, we will use sci-kit learn to implement the various machine learning algorithms. Tree-tagger is an open-source decision-tree library commonly used for lemmatization including annotation and part of speech classification. A critical assessment of parameters used in tree-tagger is a suitable target for improvement. After we have selected the exact algorithms, we will use their specific hyperparameters as our population. This will be represented by a list of numbers and strings, where each index in the list will represent its corresponding hyperparameter.

As a means to evaluate the child population for comparison with an existing created population, a simple evaluation with low overhead will be developed using an accuracy metric in which we measure how well each machine learning algorithm performs in terms of accurately classifying a sample with the specified hyperparameters. Accuracy, feature importance, practical application, and computational complexity will be considered in the fitness of the results.

References

Aghdam, M. H., Ghasem-Aghaee, N., & Basiri, M. E. (2009). Text feature selection using ant colony optimization. *Expert systems with applications*, *36*(3), 6843-6853.

Baayen, R. H., Chuang, Y. Y., Shafaei-Bajestan, E., & Blevins, J. P. (2019). The discriminative lexicon: A unified computational model for the lexicon and lexical processing in comprehension and production grounded not in (de) composition but in linear discriminative learning. *Complexity.*

Buchanan, E. M., De Deyne, S., & Montefinese, M. (2019). A practical primer on processing semantic property norm data [Preprint]. <https://doi.org/10.31219/osf.io/qb8en>

Bungum, L., & Gamback, B. (2010). Evolutionary algorithms in natural language processing. In *Norwegian Artificial Intelligence Symposium, Gjøvik (22).*

Gleim, R., Eger, S., Mehler, A., Uslu, T., Hemati, W., Lücking, A., ... & Hoenen, A. (2019). Practitioner’s view: A comparison and a survey of lemmatization and morphological tagging in German and Latin. *Journal of Language Modelling*, *7*(1), 1-52.

Landset, S., Khoshgoftaar, T. M., Richter, A. N., & Hasanin, T. (2015). A survey of open source tools for machine learning with big data in the Hadoop ecosystem. *Journal of Big Data, 2*(1), 24.

Lee, J. H., Lim, S., & Ahn, C. W. (2019). Automotive ECU data-based driver’s propensity learning using evolutionary random forest. *IEEE Access(7),* 51899-51906.

Nivre, J. (2015). Towards a universal grammar for natural language processing. CICLing.

Oliphant, T. E. (2006). A guide to NumPy (Vol. 1). Trelgol Publishing USA.

McKinney, W. (2010). Data structures for statistical computing in Python. *Proceedings of the 9th Python in Science Conference*, 51-56.

Pedregosa et al*.*, (2011). Scikit-learn: Machine learning in Python. Journal of Machine Learning Research (12), 2825-2830.

Schmid, H. (1994). Probabilistic part-of-speech tagging using decision trees. In *International Conference on New Methods in Language Processing*, Manchester, UK, 44-49.

Schmid, H. (1999). Improvements in part-of-speech tagging with an application to German. In Natural language processing using very large corpora. *Dordrecht, Springer,* 13-25.

Schmidt, M., Safarani, S., Gastinger, J., Jacobs, T., Nicolas, S., & Schülke, A. (2019). On the Performance of Differential Evolution for Hyperparameter Tuning. arXiv preprint arXiv:1904.06960.

Keshavarz, H., & Abadeh, M. S. (2017). ALGA: Adaptive lexicon learning using genetic algorithm for sentiment analysis of microblogs. *Knowledge-Based Systems*, *122*, 1-16.

Khan, A., Baharudin, B., Lee, L. H., & Khan, K. (2010). A review of machine learning algorithms for text-documents classification. *Journal of advances in information technology*, *1*(1), 4-20.

Uğuz, H. (2011). A two-stage feature selection method for text categorization by using information gain, principal component analysis and genetic algorithm. *Knowledge-Based Systems*, *24*(7), 1024-1032.

Song, F., Guo, Z., & Mei, D. (2010). Feature selection using principal component analysis. In *2010 international conference on system science, engineering design and manufacturing informatization* (Vol. 1, pp. 27-30). Institute of Electrical and Electronics Engineers.