УДК 004.891.2
DOI: 10.26102/2310-6018/2023.40.1.028

Применение машинного обучения для определения порядка прилагательных в английском языке

А.Д. Терехова✉, Г.В. Терехов, О.А. Сычев

Волгоградский государственный технический университет, Волгоград, Российская Федерация
nastyakr@list.ru✉

Резюме. В статье рассматривается способ решения задачи упорядочивания прилагательных в предложении на английском языке путем определения их гиперонимов. Определение гиперонима можно свести к задаче классификации, поэтому в данной работе произведено сравнение наиболее популярных методов классификации в машинном обучении: метод поиска ближайших соседей, логистическая регрессия, классификатор дерева решений, метод опорных векторов и наивный байесовский метод. Модели были обучены на выборке, содержащей прилагательные и их гиперонимы. Для анализируемого прилагательного отбираются схожие уже классифицированные прилагательные из обучающей выборки и на основе этих данных определяется наиболее семантически подходящий гипероним. Информацию о схожести слов предлагается брать из готовых эмбеддингов GloVe. Используя технику gridsearch, были подобраны оптимальные значения гиперпараметров для метода поиска ближайших соседей K-Nearest Neighbors. С помощью метрик точности (precision), полноты (recall) и F1-меры было проанализировано качество классификации данных при использовании каждого из перечисленных выше методов. Так как готовых датасетов, состоящих из классифицированных прилагательных, на данный момент нет, то для измерений вручную было классифицировано 300 прилагательных.

Ключевые слова: порядок прилагательных, обработка естественного языка, векторное представление слов, GloVe, методы классификации, гиперонимы.

Для цитирования: Терехова А.Д., Терехов Г.В., Сычев О.А. Применение машинного обучения для определения порядка прилагательных в английском языке. *Моделирование, оптимизация и информационные технологии*. 2023;11(1). URL: https://moitvivt.ru/ru/journal/pdf?id=1301 DOI: 10.26102/2310-6018/2023.40.1.028

Application of machine learning for adjective ordering in English sentences

A.D. Terekhova✉, G.V. Terekhov, O.A. Sychev

Volgograd State Technical University, Volgograd, Russian Federation
nastyakr@list.ru✉

Abstract. The article presents a methodology for solving the adjective ordering problem in English sentences by determining their hypernyms. The determining of a hypernym can be represented as a classification task; therefore, the most popular machine-learning classification methods were compared, they include the following: nearest neighbors method, logistic regression, decision classifier, support vector machine and naive Bayes method. The models were trained on a sample that contained adjectives and their hypernyms. For each adjective, similar adjectives from the training sample were selected; the most semantically appropriate hypernym was determined based on them. The use of information about word similarity from GloVe embeddings is proposed. The optimal values of hyperparameters for the K-Nearest Neighbors method were selected by means of the gridsearch technique. The quality of data classification was evaluated applying the metrics of precision, recall, and F1-measure for each of the

methods. Since there were no ready-made datasets of classified adjectives, 300 adjectives were classified manually to create necessary samples.

Keywords: adjective ordering, natural language processing, word vector representation, GloVe, classification methods, hypernyms.

For citation: Terekhova A.D., Terekhov G.V., Sychev O.A. Application of machine learning for adjective ordering in English sentences. *Modeling, Optimization and Information Technology*. 2023;11(1). URL: https://moitvivt.ru/ru/journal/pdf?id=1301 DOI: 10.26102/2310-6018/2023.40.1.028 (In Russ.).

Введение

В настоящее время автоматический анализ информации на естественном языке приобретает популярность в различных сферах человеческой деятельности, таких как маркетинг, психология, социология и прочее. Особую популярность среди естественных языков имеет английский язык, который является языком мирового общения.

Правил грамматики английского языка достаточно много, в связи с чем автоматизация их проверки является сложной задачей. Если при чтении или аудировании текста знание правил порядка (кроме основных) слов второстепенно, так как часто смысл можно определить без него, то при составлении собственных предложений соблюдение порядка слов иностранного языка представляет проблему. Поскольку внимание сосредоточено на смысле предложения, навык правильного порядка слов должен быть доведен до автоматизма путем длительной тренировки. Однако время учителя на проверку подобных упражнений ограничено. В этом случае помогает использование интеллектуальных обучающих систем, реализованных в виде онлайн-приложений [1]; строгость правил английского языка позволяет использовать методы формального моделирования на уровне понимания таксономии Блума для обучения ему [2].

Практическим примером является задача усвоения порядка прилагательных в предложении на английском языке, когда два и более прилагательных относятся к одному существительному. Порядок прилагательных зависит от гиперонима (более общей группы), в которое входит то или иное прилагательное. Например, фраза “the small red ball” звучит гораздо более естественно, чем “the red small ball”.

В большинстве источников выделяют 10 гиперонимов для прилагательных, которые должны идти в следующем порядке:

1) прилагательные, описывающие мнение об объекте (unusual, beautiful);
2) размер объекта (small, tall);
3) физические качества (thin, rough);
4) форма (round, square);
5) возраст (old, youthful);
6) цвет (blue, magenta);
7) происхождение (Dutch, Japanese);
8) материал (metal, wood);
9) тип (general-purpose, four-sided);
10) конкретная цель (hammering, cooking) [3].

Следовательно, для определения правильного порядка прилагательных в предложении необходимо определить гипероним, к которому относится каждое конкретное прилагательное.

Целью данной работы является разработка автоматизированного подхода для упорядочивания прилагательных в предложении на английском языке. Для этого ставится задача определения наиболее эффективного метода классификации прилагательных по гиперонимам.

Классификация прилагательных

Прежде чем определить правильный порядок прилагательных в тексте, необходимо произвести токенизацию текста (разделения текста на составляющие), где в качестве токена будет выступать отдельное слово, и проставить теги словам в соответствии с их частями речи (Part-of-speech – POS tagging). Это позволит выделить прилагательные в тексте для дальнейшей работы с ними. С задачей токенизации текста и расстановки тегов успешно справляются две библиотеки NLP (Natural Language Processing), получившие широкое распространение: NLTK [4] и SpaCy [5] (модель en_core_web_sm). Библиотека NLTK является самой известной библиотекой и имеет широкий спектр возможностей, была создана учеными и исследователями как инструмент, помогающий решать сложные задачи по анализу языка. Однако SpaCy значительно превосходит NLTK в производительности [6] и имеет более удобный API, что лучше подходит для разработчиков и полностью решает задачу токенизации текста и расстановки тегов [7].

Существует множество лексических баз данных английского языка, наиболее популярная из них – WordNet, разработанная в Принстонском университете США [8]. Она является общедоступной, а её структура в виде синонимических рядов, так называемых “синсетов”, позволила получить широкую популярность в задачах обработки естественного языка. В случае, когда слово имеет несколько значений, оно входит в несколько “синсетов”. Однако получить гиперонимы для прилагательных в том же виде, в котором они необходимы для определения порядка прилагательных довольно затруднительно, поэтому для нашей задачи WordNet не подходит.

Задачу определения гиперонима для прилагательных можно свести к задаче отнесения объекта к одному из заранее определенных классов на основании его признаков. Классификацию предлагается проводить на основании семантической близости слов. В качестве источника данных о семантической близости слов, на основе которого метод классификации определит прилагательное к тому или иному классу, можно использовать языковую модель, в которой слова соответствуют векторам – численным представлениям с сохранением семантической связи, построенным на основе корпусов текстов. Можно выделить два самых распространенных метода представления текста в виде последовательности векторов, основанных на нейросетях – это Word2Vec от компании Google и GloVe [9] от Стэнфордского университета. Оба этих метода показали свою эффективность при решении таких задач, как анализ тональности текстов, кластеризация текстов, поиск парафраз и т. д. Однако Word2Vec полагается только на контекстную статистику, при этом частота совместной встречаемости слов не имеет большого значения, GloVe же учитывает совместную встречаемость, векторы слов группируются по глобальной схожести, кроме того GloVe опережает Word2Vec на большинстве бенчмарков [10]. Поэтому для решения задачи определения гиперонима для прилагательного предпочтительнее использовать модель GloVe.

Существуют готовые предварительно обученные с помощью алгоритма GloVe вектора слов, доступные на сайте Стенфордского университета. В качестве обучающей выборки, на основе которой методы смогут классифицировать прилагательные, предлагается использовать заранее заготовленный словарь прилагательное – гипероним, где прилагательное представлено в виде вектора из датасета с готовыми векторами слов. Важно, чтобы в такой выборке присутствовали прилагательные для всех гиперонимов. Для определения гиперонима конкретного прилагательного необходимо попытаться определить закономерность между признаками слов с помощью методов классификации.

В целях определения метода, наиболее точно проводящего классификацию для конкретной задачи, предлагается сравнить с помощью метрик точности (precision), полноты (recall) и F1-меры следующие методы классификации: метод поиска ближайших соседей K-Nearest Neighbors (kNN) [11], наивный байесовский метод (Naive Bayes) [12], метод опорных векторов (SVM) [13], метод логистической регрессии [14] и метод деревьев решений [15].

Подробнее стоит остановиться на методе поиска ближайших соседей. Для его использования требуется обучить kNN на выборке с готовыми векторами прилагательных, на основе которой kNN сможет определить ближайших соседей. Алгоритм kNN вычисляет расстояние от конкретного прилагательного до каждого из прилагательных обучающей выборки (уже маркированных каким-либо классом), которое зависит от семантической близости слов. Из этих расстояний отбирается определенное количество прилагательных k, расстояния до которых минимальны и далее, по наиболее часто встречающемуся среди отобранных прилагательных классу, предсказывается искомый для конкретного прилагательного гипероним. Данный алгоритм позволяет произвольно настраивать количество прилагательных k, а также некоторые другие гиперпараметры, поэтому необходимо было провести предварительный вычислительный эксперимент, в ходе которого путем сравнения точности предсказаний были подобраны оптимальные значения гипепараметров.

Вычислительный эксперимент по подбору гиперпараметров для метода kNN

Для достижения максимальной точности при классификации нужно не только обучить модель kNN на качественных данных, но и правильно настроить гиперпараметры. В kNN гиперпараметрами выступают:

1) n_neighbors – количество соседей, используемых по умолчанию для запросов kneighbors;

2) weights – весовая функция, используемая в предсказании. Возможные значения: “uniform” – однородные веса, все точки в каждой окрестности имеют одинаковый вес. “distance” – взвешивание точек обратно пропорционально их расстоянию;

3) p – степень для метрики Минковского. При p = 1 для расчета расстояния используется манхэттенское расстояние, при p = 2 евклидово расстояние;

4) metric – метрика расстояния. Возможные значения: "minkowski", "manhattan", "euclidean", "chebyshev" и другие. По умолчанию используется метрика Минковского, которая в нормированном векторном пространстве, которую можно рассматривать в качестве обобщения как евклидова расстояния, так и манхэттенского расстояния;

5) algorithm – алгоритм, используемый для вычисления ближайших соседей: “auto”, “ball_tree”, “kd_tree”, “brute”. При использовании "ball_tree" и "KD_tree" расстояние между примерами хранятся в дереве, что ускоряет нахождение ближайших соседей. В случае “brute” ближайшие соседи для каждого тестового примера считаются перебором обучающей выборки. По умолчанию используется “auto”, при котором алгоритм подбирается автоматически в зависимости от обучающего набора данных;

6) leaf_size – порог переключения на полный перебор в случае выбора BallTree или KDTree для нахождения соседей;

7) n_jobs – количество параллельных заданий для поиска соседей. По умолчанию n_jobs = 1 [11].

Для оптимизации гиперпараметров мы использовали технику gridsearch, которая осуществляет подбор оптимальных значений гиперпараметров путем полного перебора

из заданного множества. Для решения задачи в данном множестве перебирались значения гиперпараметров из множеств, представленных на Рисунке 1.

```
k_range = list(range(1,31))
weight_options = ["uniform", "distance"]
leaf_size=[10,20,30,40,50,60]
p=[1,2,3,4,5]
algorithm=['auto', 'ball_tree', 'kd_tree', 'brute']
```

Рисунок 1 – Множество гиперпараметров для gridsearch
Figure 1 – Set of hyperparameters for gridsearch

Для выбора лучшего сочетания указанных параметров для каждого уникального набора будет проведена 10-кратная кросс-валидация в соответствии с техникой gridsearch. В ходе эксперимента был создан тестовый датасет, содержащий 300 прилагательных и их корректные гиперонимы (по 30 прилагательных для каждого гиперонима), для которого измерялось значение F1-меры при различных значениях n_neighbors, weights, leaf_size, p и algorithm. Использование F1-меры в данном случае является корректным, поскольку количество прилагательных для каждого гиперонима одинаковое. Фрагмент результата оценки классификации прилагательных по гиперонимам представлен в Таблице 1.

Таблица 1 – Фрагмент оценки эффективности классификации прилагательных метода kNN при различных значениях гиперпараметров
Table 1 – Fragment of kNN performance evaluation for adjective classification at different values of hyperparameters

algorithm	leaf_size	n_neighbors	p	weights	F1-score
auto	**10**	**2**	**2**	**uniform**	**0,92**
auto	40	15	5	distance	0,82
ball_tree	50	11	3	uniform	0,71
kd_tree	20	12	5	uniform	0,69
brute	10	5	3	distance	0,9

По Таблице 1 видно, что лучшее значение F1-меры, составляющее 0,92, удается достичь при значениях algorithm = auto, leaf_size = 10, n_neighbors = 2, p = 2, weights = uniform. В целом, в зависимости от гиперпараметров разброс значений F1-меры превышает 30 %.

Вычислительный эксперимент по определению наиболее эффективного метода классификации прилагательных

Для определения лучшего метода классификации прилагательных по гиперонимам возьмем тестовую и тренировочную выборки, где тренировочная выборка составляет 75 % случайно выбранных прилагательных от исходного датасета. Мы сравнивали следующие методы: метод поиска ближайших соседей, логистическая регрессия, классификатор дерева решений, метод опорных векторов и наивный

байесовский метод. При этом, в соответствии с результатами вычислительного эксперимента по определению оптимальных гиперпараметров для kNN, метод будет запущен с наиболее оптимальными настройками. Полученные результаты описаны в Таблице 2.

Таблица 2 – Сравнение методов классификации прилагательных по гиперонимам
Table 2 – Comparison of adjective classification methods by hypernyms

Метод классификации	precision	recall	F1-score
Наивный байесовский классификатор	0.96	0.95	0.95
Метод опорных векторов	**0.97**	**0.96**	**0.96**
Метод ближайших соседей	0.93	0.92	0.92
Логистическая регрессия	0.96	0.95	0.95
Классификатор дерева решений	0.88	0.87	0.87

Обсуждение результатов

Из Таблицы 2 видно, что по каждому из показателей метрик точности, полноты и F1-меры, несмотря на то что kNN был запущен с наиболее оптимальными гиперпараметрами, лучший результат дал метод опорных векторов. Таким образом, применение данного метода дает возможность достичь точности 0,97. Это может быть связано как с особенностями предметной области, так и с размерами тестовых выборок.

После того, как задача определения принадлежности прилагательного гипероним решена, непрерывный массив прилагательных из предложения сортируется в соответствии с эталонным порядком гиперонимов, после чего возвращается предложение с корректным порядком прилагательных. Это позволяет проверять ответы обучаемых и выдавать сообщения, поясняющие сделанные ими ошибки.

Заключение

В работе рассмотрен подход для определения порядка прилагательных в предложении на английском языке путем определения их гиперонимов. Предложение разбивается на токены и размечается тегами с использованием SpaCy. Предлагается использовать метод опорных векторов, обученный на выборке из прилагательных и их гиперонимов, который показал лучшее значение F1-меры 0,96. Для каждого прилагательного из предложения отбираются схожие прилагательные из обучающей выборки и на основе этих данных определяется наиболее семантически подходящий гипероним. Информацию о схожести слов предлагается брать из готовых эмбеддингов GloVe.

Планируется дальнейшее изучение влияния обучающей выборки на точность работы метода опорных векторов и применение разработанного подхода при оценке правильности порядка прилагательных в учебных задачах на английском языке.

СПИСОК ИСТОЧНИКОВ

1. Mitrovic A., Koedinger K.R., Martin B. A comparative analysis of cognitive tutoring and constraint-based modeling. *Lecture Notes in Computer Science*. 2003;2702:313–322. DOI: 10.1007/3-540-44963-9_42.
2. Углев В.А., Сычев О.А., Аникин А.В. Интеллектуальный анализ цифрового следа при оценке контрольно-измерительных материалов для поддержки принятия

решений в образовательном процессе. *Журнал Сибирского федерального университета. Техника и технологии.* 2022;15(1):121–136. DOI: 10.17516/1999-494X-0378.
3. Malkani N. A *Comprehensive guide on General English for competitive examinations*. Agra, Oswal Publishers; 2020. 518 p.
4. Yogish D., Manjunath T. N., Hegadi S.R. Review on natural language processing trends and techniques using NLTK. *Recent Trends in Image Processing and Pattern Recognition*. 2018;1037:589–606. DOI: 10.1007/978-981-13-9187-3_53.
5. Bird S, Klein E, Loper E. *Natural language processing with Python: analyzing text with the natural language toolkit*. O'Reilly Media, Inc; 2009. 502 p.
6. Cheng X., Kong X., Liao L., Li B. A combined method for usage of NLP libraries towards analyzing software documents. *Advanced Information Systems Engineering. CAiSE 2020. Lecture Notes in Computer Science.* 2020;12127:515–529. DOI: 10.1007/978-3-030-49435-3_32.
7. Sarkar D. *Text Analytics with Python: A Practitioner's Guide to Natural Language Processing*. New York, Apress; 2019. 698 p.
8. Fellbaum C. *WordNet: an Electronic Lexical Database*. Cambridge, MIT Press; 1998. 422 p. DOI: 10.7551/mitpress/7287.001.0001.
9. Pennington J., Socher R., Manning C.D. Glove: Global vectors for word representation. *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP).* 2014:1532–1543. DOI: 10.3115/v1/D14-1162.
10. Daniel T.L., Chantal D.L. *Discovering knowledge in data: an introduction to data mining.* New Jersey, Wiley-interscience. John Wiley & Sons, Inc; 2005. 222 p.
11. Haneen A.A.A., Ahmad B.A.H. Effects of distance measure choice on K-nearest neighbor classifier performance: a review. *Big Data*. 2019:221–248
12. Li B. Importance weighted feature selection strategy for text classification. *International Conference on Asian Language Processing (IALP).* 2016:344–347.
13. Cristianini N., Shawe-Taylor J. *An introduction to support vector machines: and other kernel-based learning methods*. Cambridge, Cambridge University Press; 2000. 204 p. DOI: 10.1017/CBO9780511801389.
14. Shafieezadeh-Abadeh S., Esfahani P.M., Kuhn D., Distributionally robust logistic regression. *Advances in Neural Information Processing Systems*. 2015:1576–1584.
15. Champandard A.J. *AI Game Development: Synthetic Creatures with Learning and Reactive Behaviors*. San Francisco, New Riders Pub; 2003. 500 p.

REFERENCES

1. Mitrovic A., Koedinger K.R., Martin B. A comparative analysis of cognitive tutoring and constraint-based modeling. *Lecture Notes in Computer Science*. 2003;2702:313–322. DOI: 10.1007/3-540-44963-9_42.
2. Uglev V.A., Sychev O.A., Anikin A.V. Data mining of digital footprint during assessment grading for intelligent decision making during learning process.. *Zhurnal Sibirskogo federal'nogo universiteta. Tekhnika i tekhnologii = Journal of Siberian Federal University. Engineering & Technologies.* 2022;15(1):121–136. DOI: 10.17516/1999-494X-0378. (In Russ.).
3. Malkani N. A *Comprehensive guide on General English for competitive examinations*. Agra, Oswal Publishers; 2020. 518 p.
4. Yogish D., Manjunath T. N., Hegadi S.R. Review on natural language processing trends and techniques using NLTK. *Recent Trends in Image Processing and Pattern Recognition*. 2018;1037:589–606. DOI: 10.1007/978-981-13-9187-3_53.

5. Bird S, Klein E, Loper E. *Natural language processing with Python: analyzing text with the natural language toolkit*. O'Reilly Media, Inc; 2009. 502 p.
6. Cheng X., Kong X., Liao L., Li B. A combined method for usage of NLP libraries towards analyzing software documents. *Advanced Information Systems Engineering. CAiSE 2020. Lecture Notes in Computer Science.* 2020;12127:515–529. DOI: 10.1007/978-3-030-49435-3_32.
7. Sarkar D. *Text Analytics with Python: A Practitioner's Guide to Natural Language Processing*. New York, Apress; 2019. 698 p.
8. Fellbaum C. *WordNet: an Electronic Lexical Database*. Cambridge, MIT Press; 1998. 422 p. DOI: 10.7551/mitpress/7287.001.0001.
9. Pennington J., Socher R., Manning C.D. Glove: Global vectors for word representation. *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP).* 2014:1532–1543. DOI: 10.3115/v1/D14-1162.
10. Daniel T.L., Chantal D.L. *Discovering knowledge in data: an introduction to data mining.* New Jersey, Wiley-interscience. John Wiley & Sons, Inc; 2005. 222 p.
11. Haneen A.A.A., Ahmad B.A.H. Effects of distance measure choice on K-nearest neighbor classifier performance: a review. *Big Data*. 2019:221–248
12. Li B. Importance weighted feature selection strategy for text classification. *International Conference on Asian Language Processing (IALP).* 2016:344–347.
13. Cristianini N., Shawe-Taylor J. *An introduction to support vector machines: and other kernel-based learning methods*. Cambridge, Cambridge University Press; 2000. 204 p. DOI: 10.1017/CBO9780511801389.
14. Shafieezadeh-Abadeh S., Esfahani P.M., Kuhn D., Distributionally robust logistic regression. *Advances in Neural Information Processing Systems*. 2015:1576–1584.
15. Champandard A.J. *AI Game Development: Synthetic Creatures with Learning and Reactive Behaviors*. San Francisco, New Riders Pub; 2003. 500 p.

ИНФОРМАЦИЯ ОБ АВТОРАХ / INFORMATION ABOUT THE AUTHORS

Терехова Анастасия Дмитриевна, магистрант, Волгоградский государственный технический университет, Волгоград, Российская Федерация.
e-mail: nastyakr@list.ru
ORCID: 0000-0001-7667-7059

Anastasia Dmitrievna Terekhova, Master's Student, Volgograd State Technical University, Volgograd, Russian Federation.

Терехов Григорий Владимирович, старший преподаватель кафедры ПОАС, Волгоградский государственный технический университет, Волгоград, Российская Федерация.
e-mail: grvlter@gmail.com
ORCID: 0000-0002-0289-1834

Grigory Vladimirovich Terekhov, Assistant Professor at Software Engineering Department, Volgograd State Technical University, Volgograd, Russian Federation.

Сычев Олег Александрович, кандидат технических наук, доцент кафедры ПОАС, Волгоградский государственный технический университет, Волгоград, Российская Федерация.
e-mail: oasychev@gmail.com
ORCID: 0000-0002-7296-2538

Oleg Aleksandrovich Sychev, Candidate of Technical Sciences, Associate Professor at Software Engineering Department, Volgograd State Technical University, Volgograd, Russian Federation.

Статья поступила в редакцию 11.01.2023; одобрена после рецензирования 09.03.2023; принята к публикации 20.03.2023.

The article was submitted 11.01.2023; approved after reviewing 09.03.2023; accepted for publication 20.03.2023.