

УДК 004.89

doi: 10.15622/rcai.2025.030

ПРИМЕНИМОСТЬ МЕТОДОВ ОЦЕНКИ КАЧЕСТВА СИНТАКСИЧЕСКОГО АНАЛИЗА К РЕЗУЛЬТАТАМ СИНТАКСИЧЕСКОГО АНАЛИЗАТОРА НА ОСНОВЕ БОЛЬШОЙ ЯЗЫКОВОЙ МОДЕЛИ

Е.Д. Шамаева (*derinhelm@yandex.ru*)

Н.В. Лукашевич (*louk_nat@mail.ru*)

Московский государственный университет
им. М.В. Ломоносова, Москва

При применении больших языковых моделей для задачи синтаксического анализа возникают принципиально новые ситуации: завершение синтаксического анализа с ошибкой и существенное изменение предложения, для которого выполняется синтаксический анализ (в том числе удаление или добавление слов). Данная статья посвящена исследованию влияния таких ситуаций на применимость стандартных методов оценки качества синтаксического анализа к результатам работы синтаксического анализатора на основе больших языковых моделей. Экспериментальная оценка проведена для синтаксического анализатора U-DepPLLaMA на тестовой выборке датасета предложений с синтаксической разметкой Taiga. Установлено, что к результатам синтаксического анализа на основе больших языковых моделей нельзя применять метод оценки на основе вычисления среднего значения метрик UAS и LAS на тестовых предложениях. Кроме того, без существенной модификации нельзя использовать стандартный алгоритм выравнивания наборов токенов. Реализация исследования доступна по адресу https://github.com/Derinhelm/parser_stat/tree/llm_taiga.

Ключевые слова: синтаксический анализ, деревья зависимости, большие языковые модели, токенизация.

Введение

Одной из классических задач обработки текстов является автоматический синтаксический анализ, в результате которого определяется синтаксическая структура предложения. В настоящее время синтаксический анализатор применяется как вспомогательный инструмент при разработке интерпретируемых методов решения таких задач, как распознава-

ние именованных сущностей [Li et al., 2024], [Vasiliev et al., 2023], [Alonso et al., 2021], [Nikolaev, 2023], оценка сложности текста [Morozov et al., 2024], перефразирование [Liu et al., 2023], выявление плагиата [Taufiq, 2023].

На качество синтаксического анализа влияет качество предшествующего этапа обработки текста, токенизации. На этапе токенизации происходит выделение токенов предложения (слов, знаков препинания и т.п.). Разбиение на токены существенно влияет не только на синтаксический анализ, но и на оценку качества синтаксического анализа.

Многие синтаксические анализаторы используют нейронные сети [Chen et al., 2014], [Andor et al., 2016], [Dozat et al., 2017]. Однако в последние годы для синтаксического анализа начали использовать и большие языковые модели (как без дообучения [Ezquerro et al., 2025], так и с дообучением [Hromei et al., 2024]). Данные анализаторы существенным образом отличаются от классических нейросетевых синтаксических анализаторов. Синтаксические анализаторы, не использующие большие языковые модели, устанавливают синтаксические отношения между токенами, выделенными из предложения на этапе токенизации. Синтаксические анализаторы на основе больших языковых моделей (Large Language Model, LLM) генерируют синтаксическую структуру предложения в текстовом виде. При этом, сгенерированное дерево зависимостей может не соответствовать исходному предложению.

В данной статье исследуется применимость стандартных методов оценки качества синтаксического анализа к результатам синтаксического анализатора на основе LLM. В качестве синтаксического анализатора на основе LLM выбран синтаксический анализатор U-DepPLLaMA, исследование проведено на тестовой выборке датасета предложений с синтаксической разметкой Taiga.

1. Методология исследования

1.1. Синтаксический анализ

Одним из широко используемых способов представления синтаксической структуры предложения является дерево зависимостей (пример дерева зависимостей приведен на рис. 1). Дерево зависимостей – это ориентированный граф вида дерево, в котором вершины соответствуют токенам предложения, а ребра – синтаксическим связям между токенами. Для каждого ребра предложения указан тип связи между соответствующими токенами. Каждому токenu соответствует ровно ¹ один главный токен (токен, от которого к данному токenu проведено ребро). Главным токеном для корня дерева является вспомогательный токен ROOT.

¹ Следовательно, общее количество ребер в дереве зависимостей равно количеству токенов в предложении (без учета вспомогательного токена).

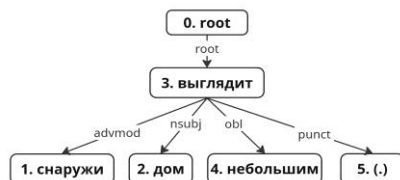


Рис. 1. Пример дерева зависимостей для предложения
«Снаружи дом выглядит небольшим»

Датасеты деревьев зависимостей используются как для обучения нейросетевых синтаксических анализаторов, так и для оценки качества их работы. Для русского языка существует несколько датасетов предложений с синтаксической разметкой. В частности, в проект Universal Dependencies² входят русскоязычные датасеты SynTagRu³, GSD⁴, PUD⁵, Taiga⁶ и Poetry⁷. Датасет Taiga основан на текстах электронной коммуникации, остальные датасеты – на литературных текстах: новостных, поэтических, публицистических и художественных. Для датасетов из проекта Universal Dependencies синтаксическая разметка производилась вручную. Кроме того, существуют датасеты, для которых синтаксическая разметка предложений выполнялась автоматически: датасеты PaRuS [Власова, 2019] и Nerus⁸, банк синтаксических деревьев RSTB^{9,10}.

Предложения из датасетов проекта Universal Dependencies и других наборов предложений с синтаксической разметкой использовались и для соревнований синтаксических анализаторов, в частности, для наиболее известных соревнований CoNLL 2017 Shared Task [Zeman et al., 2017], CoNLL 2018 Shared Task [Zeman et al., 2018], GramEval 2020 Shared Task [Ляшевская, 2020]. В этих соревнованиях принимали участие только классические нейросетевые синтаксические анализаторы.

На соревнованиях для оценки качества синтаксических анализаторов используются такие метрики, как UAS (Unlabeled Attachment Score) и LAS (Labeled Attachment Score), MLAS (Morphology-aware Labeled Attachment

² <https://universaldependencies.org/>.

³ https://universaldependencies.org/treebanks/ru_syntagrus/index.html.

⁴ https://universaldependencies.org/treebanks/ru_gsd/index.html.

⁵ https://universaldependencies.org/treebanks/ru_pud/index.html.

⁶ https://universaldependencies.org/treebanks/ru_taiga/index.html.

⁷ https://universaldependencies.org/treebanks/ru_poetry/index.html.

⁸ <https://github.com/natasha/nerus>.

⁹ <http://otipl.philol.msu.ru/~soiza/testsynt/files/info.htm>.

¹⁰ Большая часть предложений в RSTB была размечена автоматически, 800 предложений – вручную.

Score) и BLEX (Bi-LEXical dependency score) [Zeman et al., 2017]. Семантически метрика UAS соответствует доле токенов предложения, для которых верно определен главный токен, метрика LAS – доле токенов, для которых верно определен главный токен и тип связи. Метрики MLAS и BLEX дополнительно учитывают морфологические характеристики токенов и корректность определения леммы соответственно.

1.2. Выравнивание наборов токенов

Вычисление метрики производится отдельно на каждом предложении: для каждого токена происходит сравнение результата синтаксического анализа с эталоном. Однако набор токенов, из которых состоит эталонное дерево зависимостей, может отличаться от набора токенов дерева зависимостей, построенного синтаксическим анализатором. Например, в датасете Taiga предложение «Режиссёр-педагог: А. Вученович.» разделено на 7 токенов: «Режиссёр», «-», «педагог», «:», «А.», «Вученович», «.», а результат работы синтаксического анализатора на основе LLM (U-DepPLLaMA) состоит из 4 токенов: «Режиссёр-педагог», «:», «А.», «Вученович.». Для решения проблемы несоответствия наборов токенов введен предварительный этап оценки качества синтаксического анализатора: выравнивание эталонного дерева зависимостей и дерева зависимостей, построенного синтаксическим анализатором.

Стандартным алгоритмом выравнивания является алгоритм, используемый для соревнования синтаксических анализаторов CoNLL-2017 [Zeman et al., 2017]. При использовании этого алгоритма два токена считаются одинаковыми, если совпадают индексы их начала и конца в исходном предложении. Сопоставление происходит за один проход слева направо по двум наборам токенов. Листинг алгоритма сопоставления наборов токенов приведен в Приложении А.

В классическом алгоритме выравнивания предполагается, что и для эталонного дерева зависимостей, и для построенного дерева зависимостей верно следующее: при объединении (с учетом порядка) токенов дерева зависимостей получается исходное предложение (с точностью до пробелов).

1.3. Применение больших языковых моделей для синтаксического анализа

Синтаксические анализаторы, основанные на большой языковой модели, сводят задачу синтаксического анализа к задаче генерации последовательности. Для этого используется текстовая запись дерева зависимостей. Например, при применении текстовой формы записи дерева зависимостей, разработанной в [Hromei, 2024], для дерева зависимостей предложения «Дети играют в прятки» будет создана текстовая запись «[root[nsubj[Дети]][играют][obl[case[v]][прятки]]]». Соответствующее дерево зависимостей приведено на рис. 2.

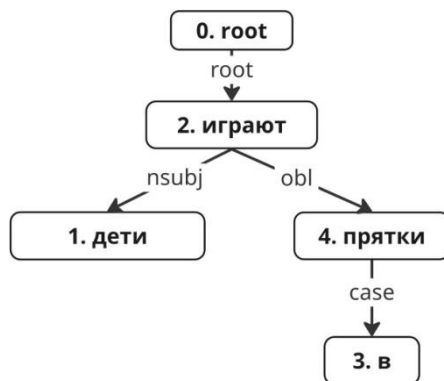


Рис. 2. Дерево зависимостей для предложения «Дети играют в прятки»

При использовании синтаксического анализатора на основе LLM в LLM передается промпт, содержащий исходное предложение. Пример такого промпта приведен в Листинге 1. В результате работы LLM возвращается исходный промпт, дополненный текстовой записью дерева зависимостей. Пример результата работы LLM приведен в Листинге 2. Далее по разработанному в [Hromei, 2024] алгоритму дерево зависимостей из текстовой записи преобразуется в стандартный формат.

Листинг 1

```

Input:
Дети играют в прятки
Answer:

```

Листинг 2

```

Input:
Дети играют в прятки
Answer:
[root[nsubj[Дети]] [играют] [obl[case[в]] [прятки]]]

```

В классических нейросетевых синтаксических анализаторах набор токенов (и соответственно вершин дерева зависимостей) определяется до начала синтаксического анализа. Этот набор токенов при объединении совпадает с исходным предложением (с точностью до пробельных символов). В синтаксическом анализаторе на основе LLM текстовое представление дерева зависимостей генерируется LLM, поэтому для некоторых предложений сгенерированный набор токенов при объединении может не совпадать с исходным предложением. Предложение, построенное по сгенерированному дереву зависимостей, может отличаться от исходного предложения. Поэтому метод сопоставления токенов на основе индексов

начала и конца токена в предложении неприменим из-за потенциального несовпадения предложений. Следовательно, для этих случаев неприменим и классический алгоритм выравнивания.

1.4. Описание эксперимента

Целью данной работы является анализ предложений, для которых неприменим классический алгоритм выравнивания. Поэтому для данного эксперимента выбрана тестовая выборка датасета Taiga, состоящего из текстов электронной коммуникации. Для этих текстов достаточно часто встречается несовпадение токенизации (за счет существенного количества опечаток, хештегов и смайликов). Тестовая выборка датасета Taiga состоит из 881 тестового предложения.

Для данной работы выбран ¹¹ синтаксический анализатор с дообучением U-DepPLLaMA ¹² [Hromei et al., 2024]. Данный анализатор построен на основе большой языковой модели LLaMA и дообучен на предложениях с синтаксической разметкой. Дообучение происходило с помощью метода LoRA с квантованием (Q-LoRA), использовались преимущественно англоязычные предложения, но присутствовали и русскоязычные. В данной статье для проведения экспериментов выбрана версия анализатора с наименьшим количеством параметров (7 миллиардов).

Для оценки качества синтаксического анализа использованы метрики UAS и LAS. Метрики MLAS и BLEX в данном исследовании неприменимы, поскольку анализатор U-DepPLLaMa не предоставляет информации о морфологических характеристиках токенов и их леммах.

2. Сравнение с существующими работами

На данный момент оценке качества синтаксических анализаторов на основе LLM посвящено небольшое количество статей. В этих статьях, в отличие от данной, не проводился детальный анализ результатов работы синтаксического анализатора. В [Hromei et al., 2024] для оценки качества синтаксических анализаторов используются только предложения, на которых синтаксический анализ завершился успешно и нет существенного несовпадения эталонного набора токенов и набора токенов, полученного в результате синтаксического анализа. В [Ezquerro et al., 2025] при оценке качества синтаксического анализа учитываются только предложения, для которых после преобразований удалось модифицировать построенное дерево зависимостей так, чтобы оно по количеству токенов совпадало с эталонным.

¹¹ Для этого анализатора в открытом доступе предоставлены и дообученные модели с разным количеством параметров, и программные реализации преобразования между графовым и текстовым форматами дерева зависимостей.

¹² <https://github.com/crux82/u-deppllama>; <https://huggingface.co/sag-uniroma2/u-depp-llama-2-7b>.

3. Результаты

При тестировании синтаксического анализатора U-DerPLLaMA на тестовой выборке датасета Taiga выявлены как предложения, для которых синтаксический анализ завершился неуспешно, так и предложения, для которых набор токенов существенным образом не совпадает с эталонным. Синтаксический анализ завершился неуспешно для 56 предложений (6.36%): LLM сгенерировала последовательность, которую невозможно корректно преобразовать в дерево зависимостей. Существенное несовпадение токенов выявлено для 87 предложений (10%). Эти предложения можно разделить на следующие группы:

- 1) с потерей токенов (слов, скобок, кавычек);
- 2) с добавлением лишних токенов;
- 3) с перестановкой токенов;
- 4) с заменой токенов.

Количество таких предложений и соответствующие примеры приведены в табл. 1 и 2. В данных таблицах используются следующие обозначения: Ист. – источник набора токенов, ЭТ – эталонный набор токенов, СА – синтаксический анализатор.

Таблица 1

Количество предложений с существенным несовпадением токенизации

Тип ошибки	Количество предложений	Доля от общего количества предложений
Потеря токенов (кроме скобок и кавычек)	10	1.14%
Лишний токен	4	0.45%
Перестановка токенов	28	3.18%
Потеря скобок	11	1.25%
Потеря кавычек	13	1.48%
Замена токена	15	1.70%

Таблица 2

Примеры существенного несовпадения токенизации

Тип ошибки	Ист.	Пример
Потеря токенов (кроме скобок и кавычек)	ЭТ	«За», «несколько», «лет», «,», «я», «видел», «множество», «постов», «с», «этими», «двумя», «песнями», «,», «теперь», «пришла», «и», «моя», «очередь», «их», «поставить», «....», «)»»
	СА	«За», «несколько», «лет», «,», «я», «видел», «множество», «постов», «с», «этими», «двумя», «песнями»
	ЭТ	«Кофе», «!», «Кофе», «!», «Кофе», «!», «Кофе», «!», «Кофе», «!», «Кофе», «!», «Кофе», «!», «Кофе», «!»
	СА	«Кофе!», «Кофе!», «Кофе!», «Кофе!», «Кофе!»

Тип ошибки	Ист.	Пример
Лишний токен	ЭТ	«Мы», «партию», «славим», «единороссов», «-», «Партию», «власти», «богатеньких», «боссов», «!»
	СА	«Мы», «партию», «славим», «единороссов», «-», «Партию», «власти», «богатеньких», «боссов!», « богатеньких »
Перестановка токенов	ЭТ	«Их», « можно », «не», «сушить», «в», «духовке», «.»
	СА	« можно », «Их», «не», «сушить», «в», «духовке.»
Потеря скобок	ЭТ	«-», «Покажи», «,», «как», «Ежик», «кушает», «яблоко», «(», «надуваем», «по», «очереди», «щечки», «)», «;», «.»
	СА	«-», «Покажи», «,», «как», «Ежик», «кушает», «яблоко», «(надуваем», «по», «очереди», «щечки», «;».
Потеря кавычек	ЭТ	«А», «кто», «там», «был», «"», «правее», «"», «,», «время», «покажет», «.»
	СА	«А», «кто», «там», «был», «"», «правее», «,», «время», «покажет.»
Замена токенов	ЭТ	« Также », «присутствует», «молодой», «Сергей», «Соседов», «.»», «#сноб_news».
	СА	« Такое », «присутствует», «молодой», «Сергей», «Соседов.»», «#сноб_news».
	ЭТ	«3», «)», «Страница», «подписана», «НАСТОЯЩИМИ», «именем», «и», « фамилией », «,», «а», «не», «вымысленными», «никнеймами», «.»
	СА	«3)», «Страница», «подписана», «НАСТОЯЩИМИ», «именем», «и», « фамилии », «,», «а», «не», «вымысленными», «никнеймами.»
	ЭТ	«Приведенные», «нами», «артикуляционные», «упражнения», «используются», « логопедами », «для», «стимуляции», «речевой», «активности», «детей», «.»
	СА	«Приведенные», «нами», «артикуляционные», «упражнения», «используются», « лагопедами », «для», «стимуляции», «речевой», «активности», «детей.»
	ЭТ	«А», « ватный », «диск», «не», «одолжите», «?».
	СА	«А», « ваттный », «диск», «не», «одолжите?».

Дополнительно выявлено, что достаточно частой ошибкой является присоединение последнего знака пунктуации к предпоследнему токеноу. Например, в предложении «Теперь пришло время для объединения.» в датасете содержится 6 токенов («Теперь», «пришло», «время», «для», «объединения», «.»), а в результате работы синтаксического анализатора – 5 токенов, поскольку токены «объединения», «.» соединены в один («Теперь»,

«пришло», «время», «для», «**объединения.**»). На данных предложениях с объединением токенов значения метрик UAS и LAS не могут быть равно 1.0 вне зависимости от качества синтаксического анализа. Количество таких тестовых предложений – 610 (69.24% от общего количества).

Объединение с предшествующим токеном достаточно часто происходит и для предпоследнего токена, если предпоследний токен является знаком пунктуации, а последний токен – смайликом или хештегом. Например, предложение «И это только начало!;)» в датасете рассматривается как 6 токенов («И», «это», «только», «**начало**», «!», «;)»), а результат синтаксического анализа содержит 5 токенов («И», «это», «только», «**начало!**», «;)»). Количество таких предложений – 17 (1.93%).

Заключение

В данной статье исследуется применимость стандартных методов оценки качества синтаксических анализаторов к синтаксическим анализаторам на основе больших языковых моделей. Для исследования использованы синтаксический анализатор на основе большой языковой модели U-DepPPLLaMA и тестовая выборка русскоязычного датасета предложений с синтаксической разметкой Taiga.

Установлено, что к результатам таких анализаторов неприменим стандартный алгоритм выравнивания наборов токенов. Причина этого заключается в существенном различии эталонного набора токенов и набора токенов, созданного синтаксическим анализатором: потери токенов, добавления лишних токенов, перестановки токенов и замены токенов на несуществующие в исходном тексте предложения.

Кроме того, неприменима оценка качества синтаксического анализа через среднее значение метрик UAS и LAS на множестве всех тестовых предложений. Вместо этого необходимо отдельно рассматривать предложения, для которых синтаксический анализ завершился неуспешно, и предложения, с существенным несовпадением токенизации (из-за которого невозможно применение алгоритма выравнивания). На остальных предложениях метрики UAS и LAS могут быть корректно вычислены.

Дополнительно установлено, что при применении синтаксического анализатора на основе большой языковой модели возможны галлюцинации большой языковой модели. Поэтому построенное дерево зависимостей может содержать токены, отсутствующие в исходном предложении.

В дальнейшем планируется проведение исследований по адаптации метрик UAS и LAS для результатов синтаксического анализа с существенным несовпадением токенизации. Кроме того, планируется более детальное исследование ошибок анализаторов (в частности, с применением методики, разработанной при создании банка синтаксических деревьев RSTB). Помимо этого, планируется провести сравнение работы больших

языковых моделей, дообученных для синтаксического анализа, и моделей без дообучения. Перспективным направлением исследований также является проверка применимости больших языковых моделей не только для грамматик зависимостей, но и для грамматик составляющих и гибридных подходов.

Список литературы

- [Власова, 2019] Власова Н.А. [и др.]. PaRuS – синтаксически аннотированный корпус русского языка // Программные системы: теория и приложения. – 2019. – Т. 10, № 4(43). – С. 181-199. – doi: 10.25209/2079-3316-2019-10-4-181-199.
- [Ляшевская, 2020] Ляшевская О.Н., Шаврина Т.О., Трофимов И.В., Власова Н.А. Grameval 2020: дорожка по автоматическому морфологическому и синтаксическому анализу русских текстов // Annual International Conference Dialogue. (Москва, 17 июня – 20 июня 2020 г.). Труды конференции. – С. 553-569. – doi: 10.28995/2075-7182-2020-19-553-569.
- [Alonso et al., 2021] Alonso M.A., Gómez-Rodríguez C., Vilares J. On the use of parsing for named entity recognition // Applied sciences. – 2021. – Vol. 11(3). – P. 1090. – doi: 10.3390/app11031090.
- [Andor et al., 2016] Andor D., Alberti C., Weiss D. et al. Globally normalized transition-based neural networks // In: Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (ACL), Berlin, Germany, 2016. – P. 2442-2452. – doi: 10.18653/v1/P16-123.
- [Chen et al., 2014] Chen D., Manning C.D. A fast and accurate dependency parser using neural networks // In: Proceedings of the Conference on Empirical Methods in Natural Language Processing, Doha, Qatar, 2014. – P. 740-750. – doi: 10.3115/v1/D14-1082.
- [Dozat et al., 2017] Dozat T., Manning C.D. Deep Biaffine Attention for Neural Dependency Parsing // International Conference on Learning Representations (ICLR). – 2017.
- [Ezquerro et al., 2025] Ezquerro A., Gómez-Rodríguez C., Vilares D. Better Benchmarking LLMs for Zero-Shot Dependency Parsing // In: Proceedings of the Joint 25th Nordic Conference on Computational Linguistics and 11th Baltic Conference on Human Language Technologies (NoDaLiDa/Baltic-HLT 2025), Tallinn, Estonia, 2025. – P. 121-135.
- [Hromei et al., 2024] Hromei C.D., Croce D., Basili R. U-DepPLLaMA: Universal Dependency Parsing via Auto-regressive Large Language Models. IJCoL // Italian Journal of Computational Linguistics. – 2024. – Vol. 10(1). – doi: 10.4000/125nm.
- [Li et al., 2024] Li L., Chen Z., Liao S. et al. Event Extraction in Complex Sentences Based on Dependency Parsing and Longformer // In: Proceedings of 2024 International Conference on Machine Learning and Intelligent Computing, Wuhan, China, 2024. – P. 1-7.
- [Liu et al., 2023] Liu T., Sun Y., Wu J. et al. Unsupervised Paraphrasing under Syntax Knowledge // In: Proceedings of the AAAI Conference on Artificial Intelligence. – 2023. – P. 13273-13281. – doi: 10.1609/aaai.v37i1.1.26558.
- [Morozov et al., 2024] Morozov D., Lagutina K., Drozhzhichikh G. et al. Exploring the Feature Space for Cross-Domain Assessing the Complexity of Russian-Language Texts // In: 2024 Ivannikov Ispras Open Conference (ISPRAS), Moscow, Russian Federation, 2024. – P. 1-8. – doi: 10.1109/ISPRAS64596.2024.10899137.

- [Nikolaev, 2023] Nikolaev I.E. Knowledge and skills extraction from the job requirements texts // *Ontology of Designing*. – 2023. – Vol. 13(2). – P. 282-293. – doi: 10.18287/2223-9537-2023-13-2-282-293.
- [Taufiq, 2023] Taufiq U., Pulungan R., Suyanto Y. Named entity recognition and dependency parsing for better concept extraction in summary obfuscation detection // *Expert Systems with Applications*. – 2023. – Vol. 217. – P. 119579. – doi: 10.1016/j.eswa.2023.119579.
- [Vasiliev et al., 2023] Vasiliev S., Korobkin D., Fomenkov S. Extracting the Component Composition Data of Inventions from Russian Patents using Dependency Tree Analysis // In: 2023 International Conference on Industrial Engineering, Applications and Manufacturing (ICIEAM), Sochi, Russian Federation, 2023. – P. 1030-1034. – doi: 10.1109/ICIEAM57311.2023.10139170.
- [Zeman et al., 2017] Zeman D., Hajic J., Popel M. et al. CoNLL 2017 Shared Task: Multilingual Parsing from Raw Text to Universal Dependencies // In: *Proceedings of the CoNLL 2017 Shared Task: Multilingual Parsing from Raw Text to Universal Dependencies*, Vancouver, Canada, 2017. – P. 1-19. – doi: 10.18653/v1/K17-3001.
- [Zeman et al., 2018] Zeman D., Hajic J., Popel M. et al. CoNLL 2018 shared task: Multilingual Parsing from Raw Text to Universal Dependencies // In: *Proceedings of the CoNLL 2018 Shared Task: Multilingual Parsing from Raw Text to Universal Dependencies*, Brussels, Belgium, 2018. – P. 1-21. – doi: <https://doi.org/10.18653/v1/K18-2001>.

Приложение А. Листинг программы стандартного сопоставления токенов

В Листинге 3 приведен псевдокод программы стандартного сопоставления токенов, разработанный в [Zeman et al., 2017]. Данный псевдокод модифицирован для иллюстративных целей¹³. В переменной `gold_words` хранится эталонный список токенов, в переменной `system_words` – список токенов, построенный синтаксическим анализатором. Для каждого токена заданы атрибуты `start` и `end` – индексы начала и конца токенов в строке-предложении. В переменной `alignment` хранятся токены, которые есть и в эталонном наборе токенов, и в наборе токенов, построенном синтаксическим анализатором.

Листинг 3

```
gi, si = 0, 0
while gi < len(gold_words) and si < len(system_words):
    if (gold_words[gi].start, gold_words[gi].end) == (system_words[si].start, system_words[si].end):
        alignment.append_aligned_words(gold_words[gi], system_words[si])
        gi += 1
        si += 1
    elif gold_words[gi].start <= system_words[si].start:
        gi += 1
    else:
        si += 1
```

¹³ Оригинальная реализация программы находится по адресу <https://universaldependencies.org/conll17/evaluation.html>.