Содержание

Аннотация

На данный момент большая часть программных задач, не решаемых алгоритмически, решается методами машинного обучения. В случае достаточно сложных задач очень часто этот метод – нейросети различной структуры. Одна из основных проблем нейросетевых решений задач – неинтерпретируемость нейросетей. Один из вариантов решения этой проблемы — probing обученной нейросетевой модели [1] — заключается в том, чтобы обучать более простые модели на предсказание понятных человеку признаков в данных по внутреннему представлению этих данных (так же называемых эмбеддингами) в нейросети.

Пробинговые эксперименты часто применяются к языковым моделям, то есть нейросетевым моделям, применяемым для различных задач, связанных с языком: машинного перевода, классификации текстов, генерации текстов, и т.д.

В данной работе рассматриваются эксперименты над языковыми моделями, связанными с признаками на уровне слов, то есть лингвистическими признаками слов, а не целых предложений (например, признаком на уровне предложения может быть количество частей, составляющих сложное предложение, или число подлежащего в предложении, а признаком на уровне слова может быть число этого слова или вид глагола).

Словарь ключевых понятий

Языковая модель - нейронная сеть, обученная для задачи работы с текстом, например, для предсказания вероятностей следующего слова в тексте.

Интерпретация нейронных сетей - методы, помогающие понять, как устроена нейронная сеть внутри, какую информацию она хранит о данных.

Эмбеддинги слов - представление данных на одном из слоёв нейронной сети.

Пробинговые эксперименты (probing experiments) - метод интерпретации нейронных сетей, состоящий из обучения модели машинного обучения для предсказания интерпретируемых человеком признаков данных по эмбеддингам данных в нейронной сети.

Эксперименты на уровне слов - пробинговые эксперименты на признаках на уровне слов, то есть относящихся не к предложению в целом, а к отдельным словам.

Контекстуализированные эксперименты. Эсперименты на уровне слов, в которых модели подаётся на вход предложение целиком, а затем достаются эмбеддинги, соответствующие интересующему нас слову.

Неконтекстуализированные эксперименты. Эксперименты на уровне слов, при которых каждое слово подаётся на вход модели отдельно от других слов.

1 Введение

1.1 Постановка задачи пробинг-эксперимента

Пробинг-эксперимент можно описать так [1]. У нас есть некоторая обученная нейронная сеть S. Пусть $S_i(x)$ - эмбеддинг наблюдения x на i-м слое нейронной сети. В нашем случае x может быть словом или предложением, $S_i(x)$ - это вектор чисел некоторой размерности, соответствующей нейронной сети.

Мы хотим узнать, хранит ли нейронная сеть S информацию о каком-то понятном человеку признаке о данных. Например, мы работаем с языковой моделью и хотим узнать, хранит ли она информацию о роде данного слова. Пусть у нас есть датасет, содержащий наблюдения - X, и вектор соответствующих этим наблюдениям целевых переменных (для нашего примера это род слова из наблюдения) - y.

Мы хотим зафиксировать некоторый слой k и для него составить новый датасет, в котором i-м наблюдением будет $S_k(X_i)$, а вектор целевых переменных будет всё тот же y. Далее мы обучим какую-нибудь модель машинного обучения (обычно для этого выбирают логистическую регрессию, рассуждения о мотивации такого выбора можно найти в [1]) на этих данных и посмотреть на качество её предсказания. Чем выше итоговое качество, тем больше мы можем быть уверены в том, что нейросеть действительно хранит эту информацию на этом слое.

Понятно, что итоговое число довольно сложно оценить - мы получили какое-то значение метрики предсказания, как понять "хорошее"оно или нет. Поэтому обычно такие метрики сравниваются с аналогичными экспериментами, например, сравниваются результаты экспериментов на разных слоях, для разных признаков, и т.д.

1.2 Описание предметной области

Есть множество статей и исследований пробинг экспериментов для признаков на уровне предложений. статья1, статья2, статья3

Эксперименты на уровне слов обделены вниманием в области пробинговых экспериментов. Про них есть статьи статья1, статья2, но их довольно мало и в них сама архитектура этиъ экспериментов довольно плохо описана. Чтобы понять, какой именно тип экспериментов проводится (в частности, являются ли эксперименты контекстуализированными), иногда приходится изучать репозитории с кодом и данными.

1.3 Постановка задачи

В рамках данной работы был разработан мини-фреймворк для проведения пробинговых экспериментов на уровне слов. Была реализована работа с данными в формате Universal Dependencies [10]. Фреймворк был спроектирован таким образом, чтобы было удобно сравнивать контекстуализированные и неконтекстуализированные пословные пробинг эксперименты.

Кроме того, были проведены различные контекстуализированные и неконтекстуализированные эксперименты, нарисованы графики зависимости от слоя качества предсказаний и количества итераций обучения логистической регрессии до сходимости. Работа проводилась с французским языком с моделью Camembert [6], но фреймворк может работать и для других языков и моделей (с оговорками, описанными в пункте 3.2.1).

2 Обзор литературы

2.1 Пробинг

Фактически единственным широко применяемым методом интерпретации языковых моделей является пробинг [8]. В качестве классификатора для пробинга можно использовать не только классические модели машинного обучения. Например, было показано, что методы пробинга, напрямую пытающиеся измерять информацию, содержащуюся в эмбеддингах нейросети, могут показывать себя стабильнее [11], чем классический пробинг логистической регрессией. Правда, они же дали неожиданные результаты при сравнении продвинутой языковой модели-трансформера Bert с простой моделью FastText [7], а именно, Bert лишь немного выигрывал у FastText. В данной работе используется классическая версия пробинга, но в дальнейшем было бы интересно добавить и версию со сравнением информации.

2.1.1 Пробинговые эксперименты на уровне предложений

Проводилось множество экспериментов с пробингом на уровне предложений, то есть пробингом для лингвистических признаков, определённых на уровне целых предложений, а не отдельных слов (например, род подлежащего, информация о структуре сложных частей предложения, и т.д.).

Создатели мультилингвальной модели Bloom [12] провели пробинг-эксперименты для многих языков (и получили, что их новая модель получает лучший accuracy почти по всем экспериментам). В этом исследовании не проводился пробинг по слоям, а результаты по всем слоям усреднялись (как мы видим в том числе из экспериментов в секции 4 данной работы, значения по слоям могут различаться).

Исследователи из Facebook AI [4] занимались пробингом нескольких языковых моделей на признаки, связанные со структурой предложения, такие как глубина дерева зависимостей в предложении, индекс подлежащего в предложении, и т.д. Так они провели исследование зависимости качества предсказаний этих признаков в зависимости от количества итераций обучения нейронной сети.

2.1.2 Пробинговые эксперименты на уровне слов

Так же есть статьи и про пробинговые эксперименты на уровне слов, то есть пробинг для лингвистических признаков, определённых на уровне слов (например, род/число слова, и т.д., см. пункт 4.1).

Например, в [2] проводятся пробинговые эксперименты моделей машинного перевода на признаки, относящиеся к частям речи или более подробным семантическим признакам. Эти эксперименты изучают зависимости качества предсказаний от слоя и от целевого языка. В этом исследовании использовался датасет семантических признаков [3]. Для целей данной работы больше подошли данные Universal Dependencies [10], поскольку они более богаты признаками и данными о зависимостях.

Кроме того, исследование пробинга контекстуализированных эмбеддингов слов для признаков уровня предложения [9] показали, что информация об удалённых словах действи-

тельно содержится в эмбеддингах других слов, хотя улучшение эффективности пробинга за счёт контекста и сильно зависит от задачи.

3 Описание реализованного фреймворка

Код открыт для публичного доступа по ссылке [5].

3.1 Сопоставление токенизации модели и Universal Dependencies

Для того, чтобы проводить эксперименты, нужно научиться доставать из эмбеддингов токенов, на которые токенизатор модели разбивает предложение, эмбеддинги интересующего нас слова, которые мы получаем из токенизации Universal Dependencies. Например, посмотрим на предложение "J'aime le camembert"(фр. "Я люблю камамбер"). Модель Camembert разбивает его на такие токены: [_J, ', aime, _le, _ca, member, t] (символом нижнего подчёркивания обозначается, что данный токен - начало нового слова). Поскольку нас интересуют только слова, а не токены, начать необходимо с того, чтобы объединить токены в слова.

К сожалению, это не решает все проблемы. Например, в токенизации Universal Dependencies предложение "Y a-t-il un risque"(фр. "Есть ли риск") токенизируется так: (Y, a, -t-il, un, risque). В токенизации же Camembert получается такой вариант: (Y, a-t-il, un, risque). Решение этой проблемы, разработанное в рамках данной работы, такое: необязательно решать эту проблему во всех случаях, главное - решить её в большинстве случаев, чтобы набрать достаточно данных для проведения экспериментов. Был разработан алгоритм, сливающий разбиение модели и разбиение universal dependencies при помощи объединения соседних токенов universal dependencies. Алгоритм реализован в функции $join\_parsings$.

3.2 Структура фреймворка

3.2.1 Извлечение эмбеддингов

Для неконтекстуализированных экспериментов используется простая функция $get\_average$, которой передаётся слово и интересующие нас слои и которая возвращает эмбеддинги на этих слоях.

Для контекстуализированных экспериментов используется функция get_word_embs, которой передаётся предложение, индексы интересующих нас слов и слоёв, и она возвращает соответствующую матрицу.

Для объединения токенов и соответствующих им эмбеддингов в слова и эмбеддинги слов используется специальный класс. Поскольку логика объединения слов может зависеть от модели (например, Camembert и Bert по-разному обозначают начало нового слова), этот класс тоже должен зависеть от модели. В рамках данной работы эксперименты проводились для модели Camembert, чтобы проводить их для другой модели, нужно дописать соответствующий класс и передавать его в функции экспериментов вместо класса CamembertMergeTokens.

3.2.2 Получение датасетов для пробинга

Для пробинга необходимы датасеты, в которых признаки - эмбеддинги слов, а целевая переменная - интересующий нас лингвистический признак. Для промежуточного этапа сборки таких датасетов есть функции *load_word_in_sentence_data* для контекстуализированных экспериментов и *load_separate_word_data* для неконтекстуализированных экспериментов. Каждая из этих функций использует специальную функцию *features_extractor*, которую они принимают в качестве аргумента и применяют к предложениям для нахождения наблюдений.

Например, если вызвать *load_separate_word_data* и передать в качестве $features_extractor$ функцию, которая находит в предложении все глаголы, возвращает список их индексов и список чисел, которые означают число глагола, то на выходе получится датасет, в котором каждая строка - глагол и число этого глагола (см картинку 3.1).

Рис. 3.1: Пример вызова команды *load_separate_word_data*. Надпись "38 out o 416 failed"показывает, на скольких предложениях алгоритм сопоставления токенизации не сработал

3.2.3 Запуск логистических регрессий

Функция *rerun_logregs* делает несколько запусков логистической регрессии на одних и тех же данных (которые передаются ей в качестве параметров). Запуски выполняются параллельно на нескольких ядрах[1] (количество ядер - один из параметров этой функции). Функция возвращает итоговое качество и количество итераций обучения логистической регрессии до сходимости на всех запусках для всех слоёв.

3.2.4 Проведение экспериментов

Самые главные функции - функции запуска экспериментов. *word_in_sentence_experiment* используется для запуска контекстуализированных экспериментов, а *separate_word_experiment* - для неконтекстуализированных. Они запускают соответственно *load_word_in_sentence_data*

[1]По какой-то причине в google colab не получается выключить multithreading для sklearn, из-за этого в google colab у меня не получилось ускорить перезапуски логистических регрессий за счёт многопоточности.

или *load_separate_word_data*, вычисляют для них эмбеддинги при помощи *get_average_embs* или *get_word_embs*, полученные данные отправляют в *rerun_logregs*, и результат возвращают.

Далее его можно передать в функцию *show_results*, которая строит *box_plot* данных.

3.3 Некоторые оптимизации производительности

3.3.1 Параллельные запуски логистической регрессии

Для того, чтобы параллельность действительно ускоряла обучение логистических регрессий, сначала пришлось выключить многопоточные вычисления в numpy. Для многопоточных вычислений был использован модуль multiprocessing.

3.3.2 Объединение нескольких наблюдений

Несколько наблюдений могут быть взяты из одного предложения. В контекстуализированных экспериментах модель для каждого наблюдения запускается на всём предложении. Если несколько наблюдений взяты из одного предложения, достаточно один раз запустить модель на этом предложении и извлечь все нужные эмбеддинги из этого одного запуска. Чтобы обеспечить возможность такой оптимизации функция *load_word_in_sentence_data* возвращает данные в таком формате, что в одной строке записано предложение и объединено несколько наблюдений, которые к нему относятся. После этого *word_in_sentence_experiment* вызывает функцию $get_word_embeddings$ и передаёт ей все уникальные индексы слов, а затем результат преобразует в формат для запуска логистических регрессий.

4 Описание экспериментов и интересные результаты

4.1 Простые пословные эксперименты

Были проведены эксперименты для различных признаков. А именно, предсказание части речи по слову, предсказание числа и рода для существительных, прилагательных и глаголов, предсказание типа местоимения, предсказание типа артикля, а также предсказание вида, лица, времени, наклонения глагола.

На 4.1 изображены результаты эксперимента для предсказания части речи для контекстуализированного случая, на 4.2 - для неконтекстуализированного случая. Можно видеть, что для нулевого слоя в обоих случаях качество примерно одинаковое (ожидаемый эффект, нулевой слой - это просто токенизация и преобразование в вектора), а далее для следующих слоёв в контекстуализированном случае значительно увеличивается качество, в неконтекстуализированном - нет. При этом в неконтекстуализированном случае качество падает к последнему слою (эта закономерность прослеживается почти во всех экспериментах). Кроме того, количество итераций обучения логистической регрессии до сходимости у обоих вариантов максимально на последнем слое, но в неконтекстуализированном случае оно в целом гораздо больше, чем в контекстуализированном.

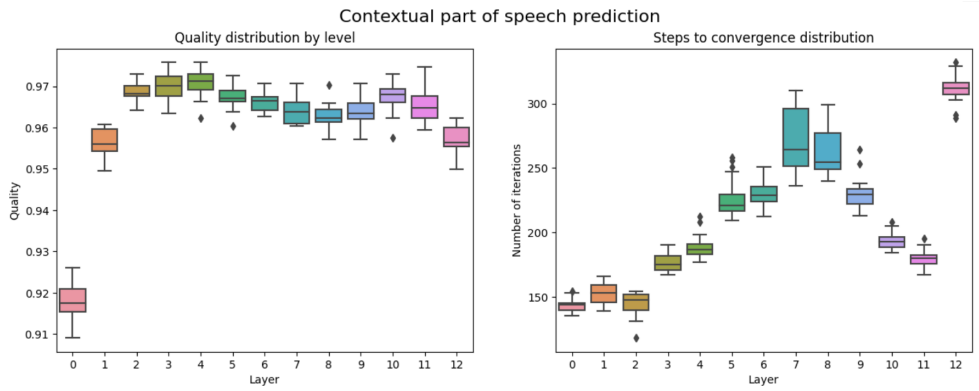

Рис. 4.1: Эксперимент по предсказанию части речи для контекстуализированного случая

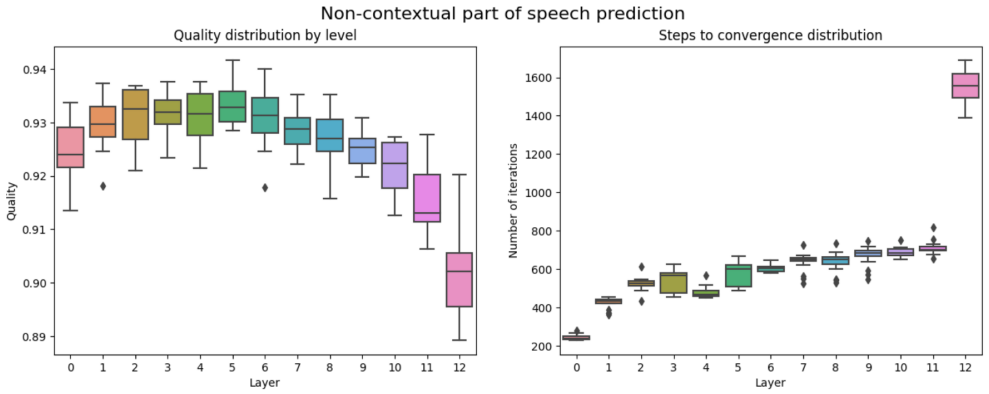

Рис. 4.2: Эксперимент по предсказанию части речи для неконтекстуализированного случая

4.2 Зависимость прилагательного от существительного

Предсказание того, зависит ли прилагательное от существительного (программа брала некоторое прилагательное, которое зависит от существительного, добавляла эту пару с целевой переменной, равной 1, а затем добавляло это же прилагательное с другим существительным с целевой переменной, равной 0).

На 4.3 изображены результаты эксперимента для предсказания зависимости прилагательного от существительного для контекстуализированного случая, на 4.4 - для неконтекстуализированного случая. Видно, что для неконтекстуализированного случая качество остаётся на уровне 0.5, что логично, потому что без контекста предсказывать зависимости невозможно. В контекстуализированном же случае качество поднимается до уровня 0.6 уже на четвертом слое и примерно там и остаётся.

4.3 Эксперименты, связанные с расстоянием

Данные эксперименты проводились только в контекстуализированном варианте. Здесь брались какие-то эксперименты из пункта 4.1 и вместо того, чтобы предсказывать признак некоторого слова по его эмбеддингу, предсказывались признаки этого слова по слову, удалённому от него на какое-то расстояние по предложению.

По картинкам 4.5-4.9 можно посмотреть на то, как изменяется качество предсказания

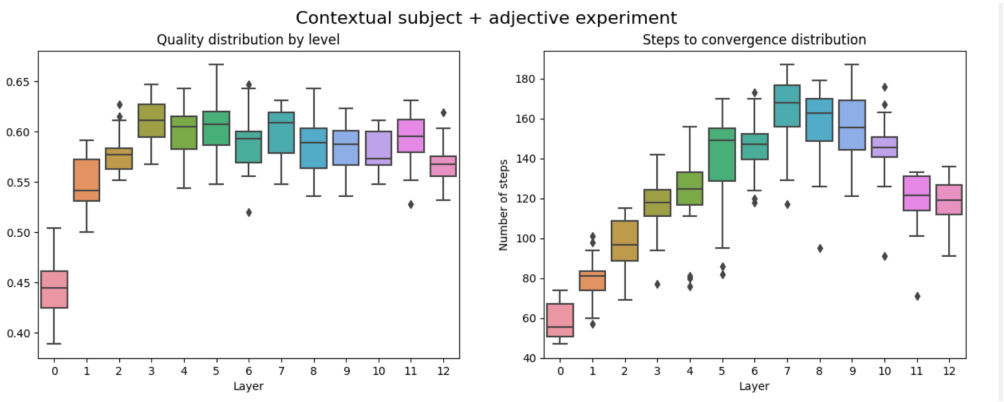

Рис. 4.3: Эксперимент по предсказанию части речи для контекстуализированного случая

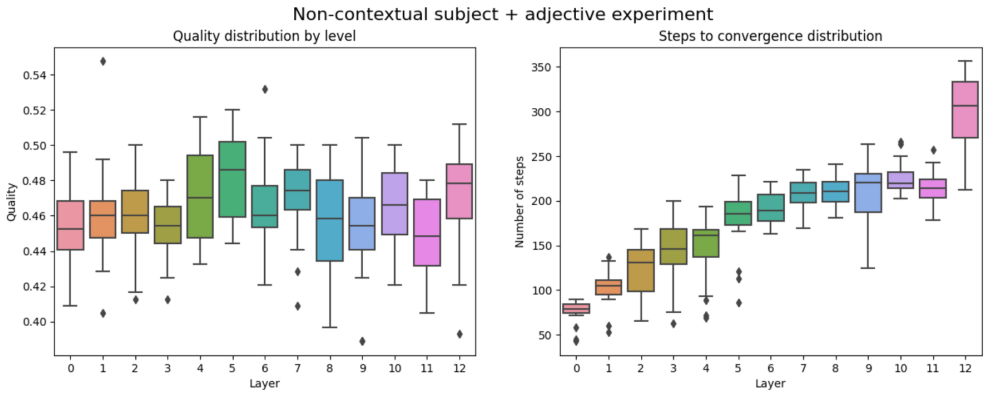

Рис. 4.4: Эксперимент по предсказанию части речи для неконтекстуализированного случая

числу по слову по мере удаления от слова. Можно видеть, что качество действительно постепенно падает на всех слоях. Интересно сравнить это с картинками 4.10 и 4.10. Здесь видно, что в отличие от случая с числом слова, качество предсказания определённости артикля очень сильно падает, если мы отступаем не на 1 слово вбок, а на два. Это можно объяснить тем, что артикль в основном имеет значение только для соседнего слова, тогда как число слова может быть связано и с более удалёнными зависимыми от него словами.

5 Выводы

5.1 Обобщённые результаты

Из общих закономерностей можно заметить, что контекст действительно явно оказывает влияние на то, какую информацию хранит нейронная сеть, и даже в случае признаков, которые имеют отношение только к конкретному слову, это влияет на качество. Кроме того, почти во всех экспериментах на последнем слое есть пик количества итераций, необходимых логистической регрессии для сходимости. Это может означать, что на последнем слое происходит что-то специальное, чего не происходит на предыдущих слоях.

В случае контекстуализированных экспериментов пик качества обычно приходится примерно на середину, а ещё очень часто есть ярко выраженный пик количества итераций

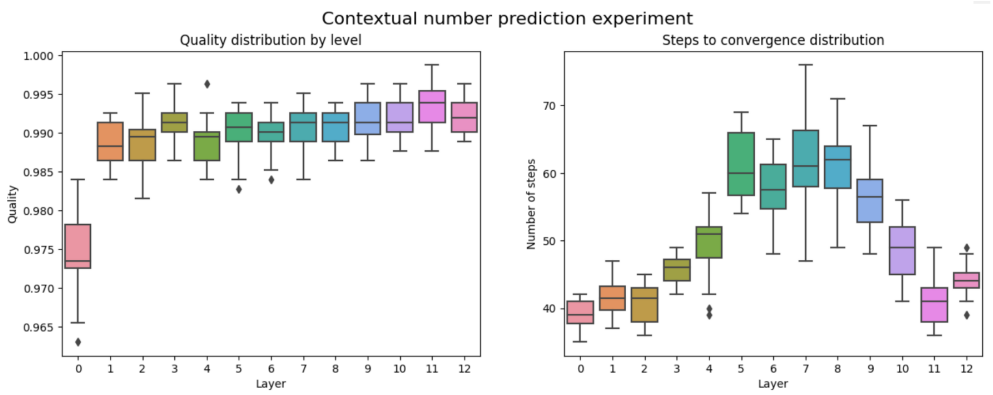

Рис. 4.5: Эксперимент по предсказанию числа по слову

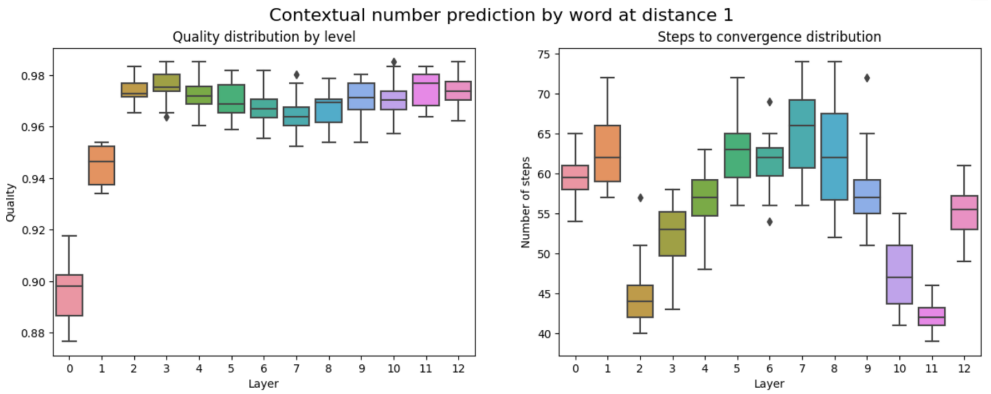

Рис. 4.6: Эксперимент по предсказанию числа по слову на расстоянии 1

до сходимости на седьмом слое. Может быть, именно там происходит в каком-то смысле пик сложности данных.

5.2 Дальнейшая работа

Для дальнейшего развития фреймворка есть несколько направлений. Во-первых, можно добавлять поддержку новых языков и моделей посредством написания новых TokenMerger классов. Во-вторых, можно оптимизировать производительность фреймворка за счёт добавления базы данных, в которую будут сохраняться эмбеддинги, чтобы не пересчитывать их лишний раз от эксперимента к эксперименту (в этой работе было принято решение не делать этого, потому что заметно дольше работают перезапуски логистической регрессии, а не вычисление эмбеддингов). В-третьих, можно оптимизировать производительность фреймворка засчёт оптимизации гиперпараметров или выкидывания лишних признаков логистических регрессий и прочих методов ускорения логистических регрессий.

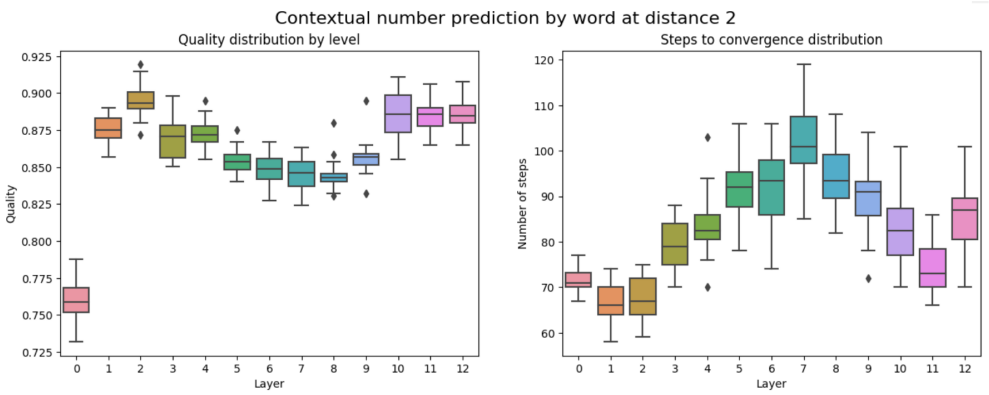

Рис. 4.7: Эксперимент по предсказанию числа по слову на расстоянии 2

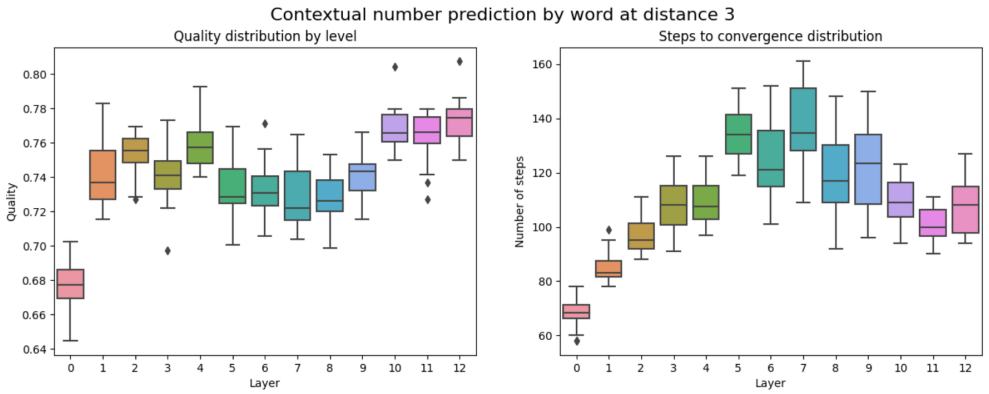

Рис. 4.8: Эксперимент по предсказанию числа по слову на расстоянии 3

Список литературы

[1] Yonatan Belinkov. *Probing Classifiers: Promises, Shortcomings, and Advances*. 2021. arXiv: 2102.12452 [cs.CL].

[2] Yonatan Belinkov, Lluis Màrquez, Hassan Sajjad, Nadir Durrani, Fahim Dalvi и James Glass. "Evaluating Layers of Representation in Neural Machine Translation on Part-of-Speech and Semantic Tagging Tasks". В: *Proceedings of the Eighth International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*. Taipei, Taiwan: Asian Federation of Natural Language Processing, нояб. 2017, с. 1—10. URL: https://aclanthology.org/I17-1001.

[3] Johannes Bjerva, Barbara Plank и Johan Bos. *Semantic Tagging with Deep Residual Networks*. 2016. arXiv: 1609.07053 [cs.CL].

[4] Alexis Conneau, German Kruszewski, Guillaume Lample, Loïc Barrault и Marco Baroni. "What you can cram into a single $&!#* vector: Probing sentence embeddings for linguistic properties". В: *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*. Melbourne, Australia: Association for Computational Linguistics, июль 2018, с. 2126—2136. DOI: 10.18653/v1/P18-1198. URL: https://aclanthology.org/P18-1198.

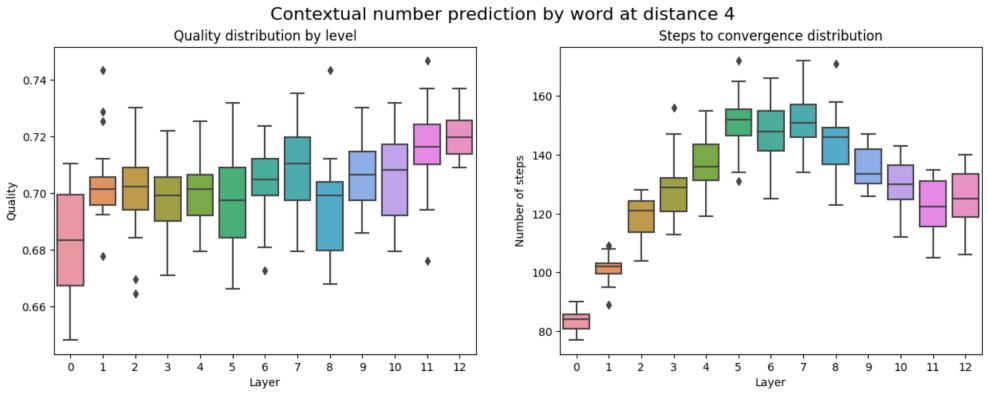

Рис. 4.9: Эксперимент по предсказанию числа по слову на расстоянии 4

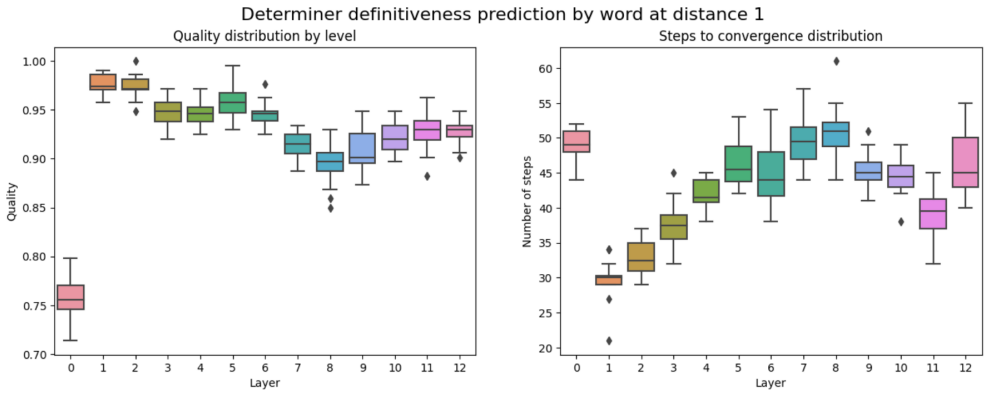

Рис. 4.10: Эксперимент по предсказанию определённости артикля по слову на расстоянии 1

[5] *Github for this project*. URL: https://github.com/yulikdaniel/WordProbing.

[6] *Hugginface camembert model*. URL: https://huggingface.co/camembert-base.

[7] Tiago Pimentel, Josef Valvoda, Rowan Hall Maudslay, Ran Zmigrod, Adina Williams и Ryan Cotterell. "Information-Theoretic Probing for Linguistic Structure". В: *CoRR* abs/2004.03061 (2020). arXiv: 2004.03061. URL: https://arxiv.org/abs/2004.03061.

[8] Ryan Teehan, Miruna Clinciu, Oleg Serikov, Eliza Szczechla, Natasha Seelam, Shachar Mirkin и Aaron Gokaslan. "Emergent Structures and Training Dynamics in Large Language Models". В: *Proceedings of BigScience Episode #5 – Workshop on Challenges & Perspectives in Creating Large Language Models*. virtual+Dublin: Association for Computational Linguistics, май 2022, с. 146—159. DOI: 10.18653/v1/2022.bigscience-1.11. URL: https://aclanthology.org/2022.bigscience-1.11.

[9] Ian Tenney, Patrick Xia, Berlin Chen, Alex Wang, Adam Poliak, R Thomas McCoy, Najoung Kim, Benjamin Van Durme, Samuel R. Bowman, Dipanjan Das и Ellie Pavlick. *What do you learn from context? Probing for sentence structure in contextualized word representations*. 2019. arXiv: 1905.06316 [cs.CL].

[10] *Universal Dependencies*. URL: https://universaldependencies.org/.

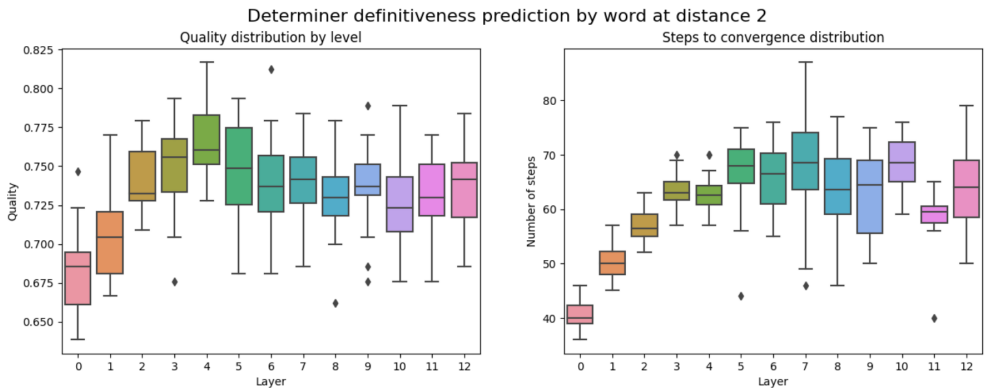

Рис. 4.11: Эксперимент по предсказанию определённости артикля по слову на расстоянии 2

[11] Elena Voita и Ivan Titov. “Information-Theoretic Probing with Minimum Description Length”. В: *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*. Online: Association for Computational Linguistics, нояб. 2020, с. 183—196. DOI: 10.18653/v1/2020.emnlp-main.14. URL: https://aclanthology.org/2020.emnlp-main.14.

[12] BigScience Workshop и др. *BLOOM: A 176B-Parameter Open-Access Multilingual Language Model*. 2023. arXiv: 2211.05100 [cs.CL].