ФЕДЕРАЛЬНОЕ ГОСУДАРСТВЕННОЕ АВТОНОМНОЕ
ОБРАЗОВАТЕЛЬНОЕ УЧРЕЖДЕНИЕ
ВЫСШЕГО ОБРАЗОВАНИЯ
«НАЦИОНАЛЬНЫЙ ИССЛЕДОВАТЕЛЬСКИЙ УНИВЕРСИТЕТ
«ВЫСШАЯ ШКОЛА ЭКОНОМИКИ»

Факультет информатики, математики и компьютерных наук

Программа подготовки бакалавров по направлению
01.03.02 Прикладная математика и информатика

Харчиков Игорь Владиславович

КУРСОВАЯ РАБОТА

Медицинская классификация на основе описания состояния пациента с использованием модели на базе BERT

Научный руководитель

Д.П. Семенов

Нижний Новгород, 2023

Содержание

Введение

Методы машинного обучения значительно помогаеют в автоматизации многиъ процессов, с некоторого времени, алгоритмы машинного и глубинного обучения стали обладать достаточными возможностями и надежностью для применения и в сфере медицины [6].

Медицина является достаточно обширной сферой как науки так и поведневной деятельности, ее аналитическая часть часто взаимодействует с большим количесвтом наблюдений разного рода, как в масштабах популяций, так и одного пациента. Даже один единственный пациент может пройти такое количество анализов, что понадобится достаточно продолжительное время и целый набор специалистов для постановки диагноза. Медицинские обследования пораждают данные в разных формах, включая, но не ограничиваяь табличными данными на разного рода тестах, визуальной информацией в в виде снимков, временных рядов при анализе разного рода биоритмов. Но еще до такого рода информации, о пациенте получают более сложно поддающуюся анализу, но легкодоступную ифнормацию - из разговор при визите ко врачу.

Из рассказа пациента о своем состоянии, можно сделать выводы о возможных заболеваниях, вычленить ключевые и формализованные симптомы, которые понадобятся для ведения истории и дальнейшего направления на анализы и ведения пациента в целом. Бывшая актуальной во все времена проблема распознавания такого рода обостряется с развитием телемедицины - то есть обращении при отсутствии очного посещения врача. Доступность пораждает больший спрос и задача формализации каждого случая становится еще острее. Более того, процесс может быть автоматизирован рекоммендательной интеллектуальной системой для дальнейшего направления, или предложения направиться на анализы, тогда и для нее понадобится формальный список аномалий в состоянии пациента.

Только недавние достижения искуственного интеллекта позволили эффективно и надежно работать с такими данными. Работа с текстами на ”человеческом языке” называется ”обработкой естественного языка” (далее - Natural Language Processing или сокращенно NLP). значительный прорыв в области обработки

такого рода данных связан с развитием глубинного обучения и дальнейшим появлением рекуррентных сетей и архитектур на основе механизма внимания. Также важно отметить проблему сокрытости зананий нейронной сети. Эти алгоритмы представляют Black Box функции, то есть неизвестен смысл их внутренних процессов. Это неприемлимо при применении в задачах реальной жизни, тем более в медецинской сфере. Поэтому требуется дополнительное исследование и интерпретация обученной модели. В случае имплементации в рабочий процесс, эксперты должны иметь возможность оценивать, как алгоритм работает с данными для валидации верности работы, для этого не достаточно обычных метрик.

Цели работы:

1. Формализация NLP-задачи для ML и обзор технологий для решения
2. Механизмы стоящие за RNN и Transformer, изучение основных механизмов и архитектур для актуальных нейронных сетей в сфере NLP
3. Осуществить файн-тюнинг на основе претренированной языковой модели BERT для решения задачи классификации на медицинских данных в виде естественного языка
4. Применяя инструменты анализа алгоритмов глубинного обучения интерпретировать связи, созданные внутри нейронной сети

1 Естественный язык как модальность данных и примененимость методов машинного обучения

1.1 Специфика репрезентации данных

Как уже было сказано, обработка естественного языка NLP одна из самых актуальных в настоящее время. Выше было отмечено, что NLP это сложный процесс, ключевыми сложностями вызванными непосредственно природой данных, являются кодирование данных и работа с непостоянной размерностью. Фраза на естественном языке - это последовательность слов некоторого языка, но для любого алоритма потребуется каким-то образом перевести ее в числовой тип, единственный принимаеиый существующими алгоритмами. Слова крайне враждебны в этом отношении, наивные методы крайне неэффективны, и, чтобы покрыть целый словарь, понадобится очень много памяти, более того, фраза даже из закодированных слов имеет нефиксированную длину. В итоге при подготовке данных для решения встает два вопроса:

- Какой метод кодирования слов оптимален в терминах сохранения информации и эффективности хранения
- Как обработать последовательность нефиксированной длины или преобразовать ее таким образом, чтобы размерность данных стала фиксированной

Последний вопрос еще более важен, так как он непосредственно связан с моделью для решения задачи.

1.2 Механизмы кодирования и репрезентации в качестве многомерных объектов

Для какой-либо задачи в области NLP будет определен некоторый конечный словарь - множество всех допустимых встречающихся слов естественного языка. Очевидным методом является применение кодирования OneHotEncoding (далее OHE), при котором i-ому слову будет соответсвовать вектор

состоящий из нулей с i-ым элементом единицей, то есть:

$$\textit{Для словаря } V, |V| = N \; \exists \; ohe : V \longrightarrow \mathbb{R}^N; ohe(V_i) = a, \forall j \neq i a_j = 0, \; a_i = 1 \quad (1)$$

Отображение 1 позволяет получить численное представление слов в виде векторов, однако размерность равна таковой у словаря.

От части эту проблему можно решить при помощи грамотного разделения фраз на слова. Вместо наивного разбиения на непосредственные слова (разделенные в языке пробелами) и знаки пунктуации можно сократить длину словаря при помощи *токенизации*. Алгоритмы токенизации могут быть разными, помимо сокращения мощности множества слов токенизация может так же сообщать полезную информацию алгоритмы, как будет показанно позже. Как самостятельный контейнер информации one-hot вектор не несет смысла, кроме явного указания на то, что за слово было закодировано. Полезным преобразованием является векторизация слова (Word To Vector - далее W2V)[1]. W2V является обучаемой системой, которая отображает токены в вектора фиксированной размерности, являющейся гиперпараметром. В итоге обучения получается матрица весов $W_{N\times S}$, алгоритм устроен таким образом, что при помощи прохода окном по послеовательности токенов, максимизируются условные вероятности появления слова из центра окна в окржении остальных слов из окна. Таким образом, выявляется контекстная значимость слов. Векторы полученные данным алгоритмом обладают удобными свойствами, в частности компактность относительно семантического значения, то есть блзкие по смысловому значению слова отобразятся в близкие векторы.

$$\textit{Для словаря } V, |V| = N \; w2v_S : V \longrightarrow \mathbb{R}^S, \; S \in \mathbb{N}; w2v_S(V_i) = ohe(v_i) \times W_{N\times S} = b \in \mathbb{R}^S \quad (2)$$

Усовершенствованием технологии является FastText [2] помимо улучшенного процесса обучения применяющий дополнительно подобие токенизации, разбивая слова на части. Это позволяет более эффективно обрабатывать неизвестные слова, так как они могут состоять из встреченных ранее токенов, и выделяет больше информации за счет работы с уровнем, ниже слов естесвенного языка. Таким образом, для работы классических методов машинного обучения можно получить векторное представление фразы в виде последовательности векторов, а затем каким либо образом преобразовать в

единственный вектор при помощи поэлементного среднего:

$$x = \frac{\sum_{i=1}^{M} v2w_S(v_i)}{M} \in \mathbb{R}^S$$

Для методов глубинного обучения инструментарий кодирования может использоваться такой же, однако для наиболее эффективных и рассмотренных далее в работе архитектур он немного отличается. Важно отметить, что методы глубинного обучения обыкновенно оперируют всей последовательностью, не сводя ее к одному вектору. Как было отмечено в [9], возможность оперировать всей последовательностью позволяет захватывать конекст последовательности, несмотря на очевидные плюсы алгоритмов, инвариантных к длине данных, они проигрывают в сфере восприятия и выделения черт (feature extraction).

Применяются достаточно продвинутые алгоритмы токенизации. BPE-токенизация [5] применяется во многих популярных и мощных решениях, алгоритм очень эффективен засчет отыскания и жадного перекодирования наиболее частых пар токенов в новый токен, не встречавшийся ранее. Как и BPE Word Piece токенизация, впервые представленныя в [7], разбивает последовательность на уровне ниже целых слов. Метод является усовершенствованной версией и вместо частоты в BPE оперирует вероятностным подходом. При выборе пар высчитывается правдоподобие - вероятность последовательного нахождения токенов друг за другом. Word Piece добавляет токены в словарь склеивая входные до тех пор, пока уровень правдоподобия не опустится до заранее выбранного порогого значения. Алгоритм так же является жадным, но в отличие от BPE итерации заданы не напрямую, а через выше обозначенный порог.

Для векторизации токенов в нейронных сетях используют так же слой вложения (Embedding). Итоговое вычисление вложенного представления схоже с 2. Блок препроцессинга включен в саму сеть и обучается вместе с ним, от обычного линейного он также отличается оптимизацией для работы с разреженными данными. При кодировании последовательности может оказаться так, что одинаковые векторы стоят на разных местах, то есть один и тот же токен может выполнять разную роль в одной фразе. Чтобы добавить информацию об относительном положении слова в последовательности, к каждому вектору добавляется значение полученное

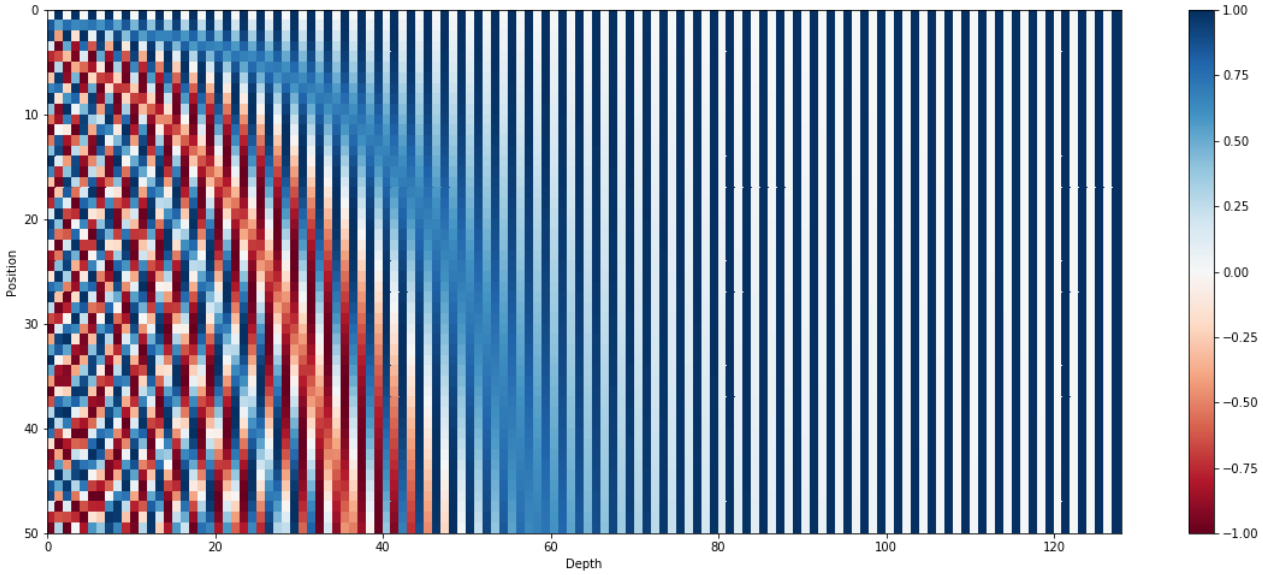
из позиционного кодирования (Positional Encoding или PE) поэлементным сложением. Значения вычисляются следующим образом [3][8]:

Пусть t - позиция в последовательности, d - размерность векторного представления, (i) - индекс элемента вектора

$$p_t^{(i)} = f(t)^{(i)} := \begin{cases} sin(w_k * t), & при\ i = 2k \\ cos(w_k * t), & при\ i = 2k+1 \end{cases}, где\ w_k = \frac{1}{1000^{2k/d}}$$

Пример полученных значений - 1.

Figure 1: Спектр значений позиционного кодирования с размерностью (Depth) 128 и максимальной длинной (Position) 50, каждая строка - вектор p_t



2 Рекуррентные сети и трансформерные архитектуры

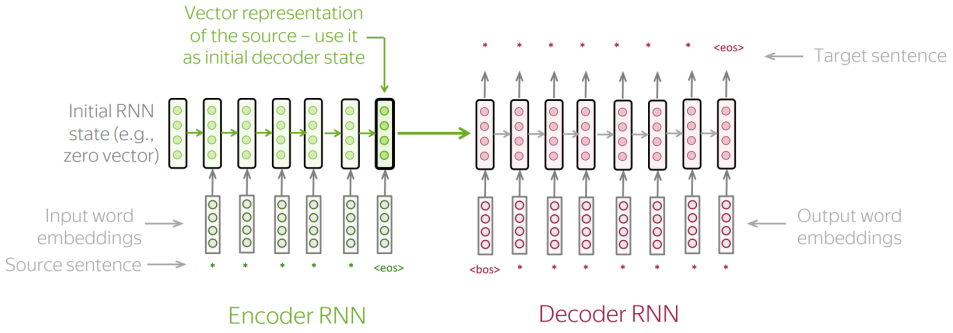
2.1 Развитие идей и инструментов обработки последовательностей

2.1.1 Энкодер-Декодер архитектура

Как было подчеркнуто в предыдущей главе, ссылаясь на [9], алгоритмы не инвариантные к длине последовательности выигрывают по многим возможностям. Скрытые Марковские Модели, или кратко СММ, относятся к такому типу алгоритмов. Однако в них состояние модели зависит только от предыдущего, хотя и существуют модификации расширяющие возможности. Рекуррентные архитектуры нейронных сетей с механизмом LSTM (Long Short Term Memory unit - Механизм Короткой Долгосрочной Памяти) предложенным в [10] обладают возможностью сохранять контекст на протяжении обработки всей последовательности. LSTM не является единственной опцией, и даже он имеет множество модификаций, однако так или иначе все рекуррентные сети отдельно обрабатывают каждый токен, но каким-либо образом учитывают информацию о предыдущих. После прохождения по последовательности вложенное представление последнего токена будет обладать информацией о всей последовательности. Более того, такое представление будет обладать все тем же свойством ”семантической компактности” как и отображение 2, однако уже в масштабе целого предложения. Помимо того, что это само по себе очень полезное свойство, на нем основано главное преумещство энкодер-декодерных архитектур на основе рекуррентных операторов.

Наиболее яркое применение этой архитектуры связано с задачей seq2seq (sequence to sequence или перевод последовательности в последовательность). Последнее соятояние энкодера используется как инициализация для состояний декодера, передавая таким образом информацию для создания новой последовательности извлеченную из входной. Эффективно, это создает ”бутылочное горлышко” для сети, заставляя ее наиболее эффективно отражать

Figure 2: Архитектура RNN для seq2seq задачи, развернутая по ходу последовательности



в этом векторе информацию из последовательности на входе.

Если задача требует не создан ие новой последовательности, но декодер заменяется на другие операторы, которые будет взаимодействовать как раз таки с последним состоянием последнего слоя энкодера, или его аналогом, как будет показано далее, как с самым информативным и полным представлением об оригинальной последовательности.

При этом итерация по последовательности может быть как в одну сторону - от начала к концу или наоборот, так и вместе, после чего вложенные преддставления будут сконкатенированны. Такой алгоритм будет называться *Двунаправленным (Bidirectional)*.

2.1.2 Применение механизма внимания

Прорывным шагом в работе с энкодер-декодерной архитектурой являлся миханизм внимания[11]. Он позволяет не просто использовать последнее состояние энкодера, но управлять тем, какие состояния предыдущего слоя и как использовать, за счет вычисления значимости внимания (Attention Scores) вычислимых между всеми элементами текущей и прошлой последовательностей.

Гладким аналагом максимума является функция *Softmax*, она и используется для получения значимости внимания:

$$A \in \mathbb{R}^S : Softmax(A_i) = \frac{A_i}{\sum_{j=1}^{S} exp(A_j)}$$

Q - Query, член для которого ведутся вычисления на данном шаге, *K - Key* - все члены последовательности, по которым итеративно идет вычесление,

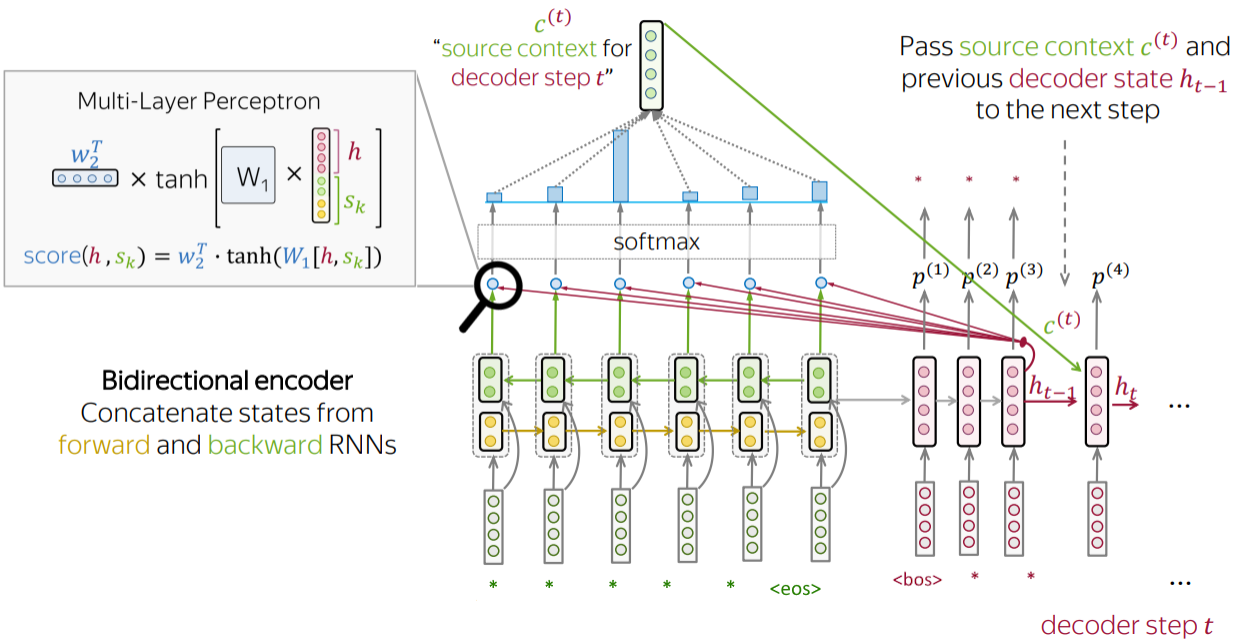
используются для получения значимости внимания между членами; *V - Value* - значение, используемое далее в вычислениях. Итоговая формула:

$$Attention(Q, K, V) = Softmax\left(QK^T\right)V \tag{3}$$

$$Здесь\ и\ далее\ d_k, d_v \in \mathbb{N}; Q, K \in \mathbb{R}^{d_k}, V \in \mathbb{R}^{d_v}$$

Применение механизма внимания отражено на 3.

Figure 3: Применение механизма внимания в двунаправленной RNN с MLP - линейными слоями, разделенными активацией-гиперболическим тангенсом,[11]



2.1.3 Трансформерная архитектура

В архитектуре трансформерной сети [3] нет обычных операторов рекуррентной сети, но используется механизм внимания с множителем для нормализации значений:

$$Attention(Q, K, V) = Softmax\left(\frac{QK^T}{\sqrt{d_k}}\right)V \tag{4}$$

При этом внимание внутри энкодера и декодера применяется с Q=K=V и называется *Self-Attention*. Между энкодером и декодером есть связующий

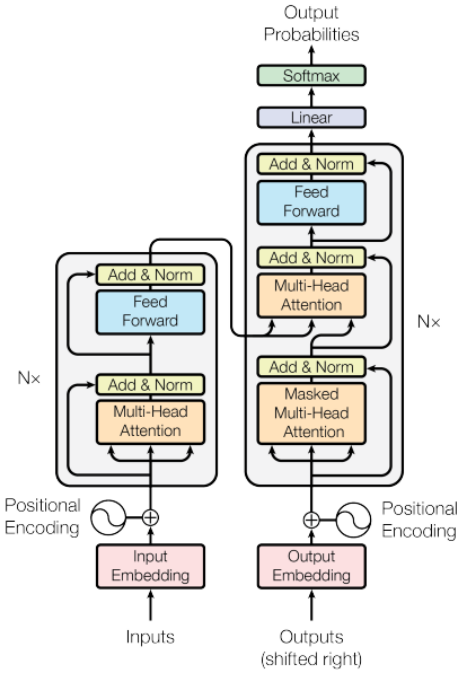
мезанизм внимания, где Q используется из декодера а K,V из энкодера 4.
Так же для больших выразительных способностей внимания, используется версия с множеством голов h:

$$MultiHeadAttention(Q,K,V) = \bigoplus_{i=0}^{h} (head_i)\, W_0, \tag{5}$$

$$head_i = Attention\left(QW_i^Q, KW_i^K, VW_i^V\right) - \textit{аналогично 4 от линейных комбинаций с собственными весами}$$

Наличие разных голов помогает выделять разные типы взаимосвязей между членами последовательности.

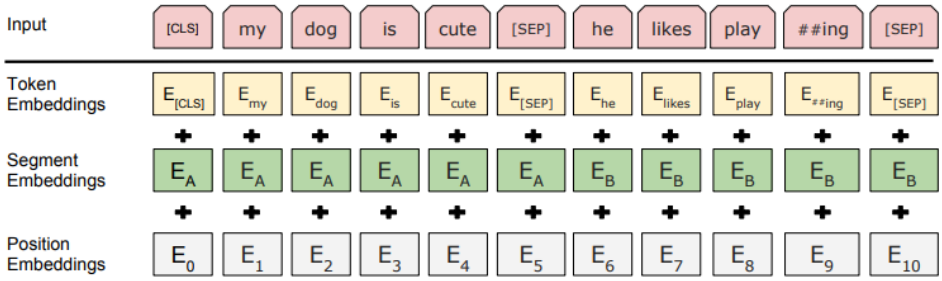
Figure 4: Архитектура трансформера



2.2 Технология обучения BERT

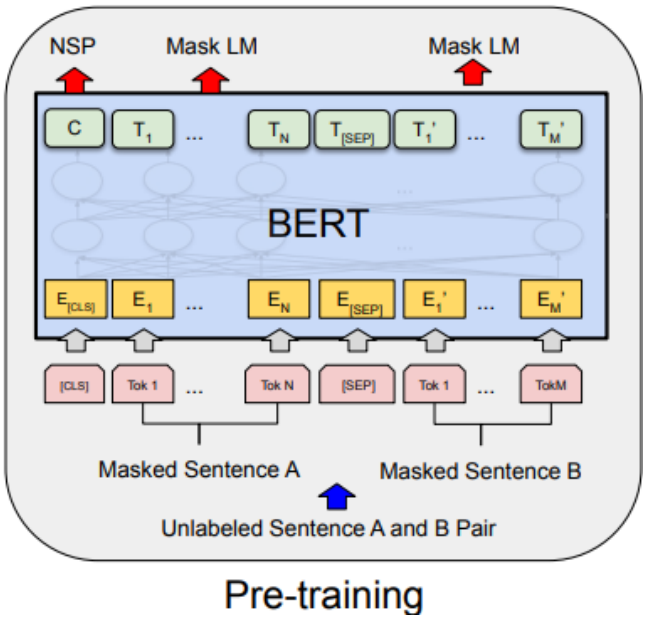
Наиболее мощная претренерованная модель трансформера представлена в [4]. При токенизации добавляются специальные сегментационные токены *[CLS]* и *[SEP]*; первый ставится в начале последовательности, второй отделяет значимые части при работе сразу с двумя последовательностями для задачи с ответом на вопрос (где и разделяетответ и вопрос). Для

Figure 5: Токенизация в BERT



обучения сети без учителя, из входных данных случайно удаляются некоторые члены, их заменяют специальным токеном *[MASK]*. Целью сети

Figure 6: Обучение без учителя с масками



становится восстановление токенов, сокрытых *[MASK]*, то есть макчимизацией вероятности выбрать правильное слова из словаря на эту позицию. Такой подход позволил обработать большое количество данных без разметки и получить очень сильную модель, которую в процессе тонкой настройки *fine-tuning* можно подготовить для любой задачи в сфере NLP. Очень мощное представление о входных данных будет содержаться в выходном *[CLS]* токене - специальном токене для классификации. Свойства представления данных в нем схоже с ”бутылочным горлышком” в энкодер-декодерных архитектурах.

3 Применение BERT к медицинским данным

3.1 Формализация задачи и обзор данных

3.1.1 Датасет NBME

NBME [12] выбран в качестве датасета для проверки алгоритма и последующей интрепретации. Задача связана с формализацией диагноза из опроса и осмотра пациента. В качестве данных представлены заметки о пациентах - отрывки наблюдений о каждом и признаки для каждого клинического случая, эти данные разделены на две таблицы.
Понятия о данных этого датасета:

- Клинический случай (clinical case): сценарий (например, симптомы, жалобы, опасения), который пациент представляет.
- Примечание пациента (patient notes): Текст, содержащий подробную важную информацию, сообщенную пациентом во время встречи (физического осмотра и собеседования).
- Признак (feature): Клинически значимая концепция. В рубрике описаны ключевые понятия, относящиеся к каждому конкретному случаю.

Обучающая и тестовые выборки разбивают данные по клиническому случаю, примечанию пациента и признаку (либо признакам), задача заключается в указании части примечания, содержащей информацию о признаке.

3.1.2 Постановка и метрика качества

Ответы представлены парами индексов $\{ i, k \}$, возможно несколько для 1 сэмпла. Как раз такие пары и должна научиться строить модель. Задача не очевидна, но ее можно свести к классификации посредствам перехода к следующему восприятию ответов: позитивным сэмплом (речь о сабсэмпле-индексе для реального сэмпла) называть входящие в обозначеный отервал индексы; по факту от модели требуется бинарно классифицировать все индексы, вместо пары $\{ i, k \}$ максимизируя вероятности для $\forall k \in [i, j)$ для верных.

Предложенная метрика для данных в соответствующем данным соревнованию - специально заданная *f1score*. Обычный вид метрики:

TP - верно предсказанный позитивный сэмпл,

FP - неверно предсказанный позитивный сэмпл,

FN - неверно предсказанный негативный сэмпл

$$Recall = \frac{TP}{TP + FN}$$

$$Precision = \frac{TP}{TP + FP}$$

$$f1score = 2 * \frac{Recall * Precision}{Recall + Precision}$$

Метрика для этой задачи имеет такую же формулу, но отличается более подходящим определением *TP, FP, FN*:

TP - индекс входит в пересечение предсказания и лейбла,

FP - входит в предсказанный, но не в лейбл,

FN - входит в лейбл, но не в предсказанный отервал

Важное свойство метрики заключается в том, что она как среднегармоническое мягко приближает минимум метрик *Полноты* и *Точности*, достаточно справедливо оценивая качество модели в терминах ошибки первого и второго рода, что вытекает из стандартного определения и сохраняется в описанном для этой задачи.

После выбора всех значащих индексов их нужно превести в вид *{ i, k }* и разделить ”;” для соответсвия требованиям при помощи несложного процесса парсинга.

3.2 Решение задачи

Для решения задачи использовался претренированая модель BERT Base Uncased и соответствующий токенайзер (работа проведена с использованием фреймверков *pytorch* и *hugging face transformers*). Поверх нее были добавлены 3 линейных слоя с дропаутом - оператором, случайно зануляющим некоторые каналы с вероятностью *p* при тренеровке и домножающим остальные значения

на $\frac{1}{1-p}$. Применение техники регуляризует сети по средствам контроля коадаптации искусственных нейронов.

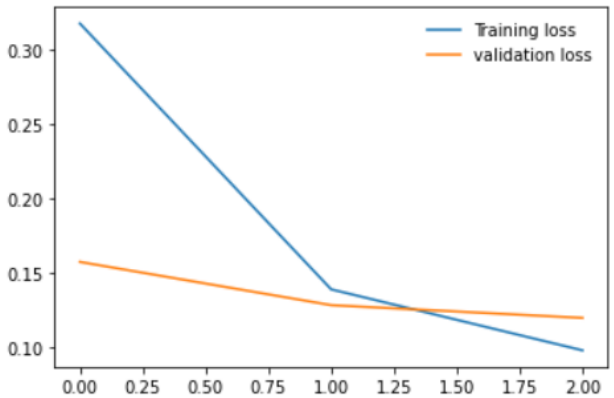
Применяя *[CLS]* токен и линейные слои получаются 1-мерные массивы, для получения вероятности требуется применить к ним фунцию $Sigmoid(x) = \frac{1}{1+e^{-x}}$. В качестве функции потерь применялась *Бинарная Кросс Энтропия*:

$$BCE(x, y) = \{l_j\}, l_j = w_n \left(y_n \times logx_n + (1y_n) \times log(1 - x_n)\right) \quad (6)$$

Для оптимизации весов в процессе обученния использовался *AdamW*, версия оптимизатора с адаптивными моментами и переработанным механизмом регуляризации на основе весов сети, файн-тюнинг длился 3 эпохи с постоянным размером шага обучения $lr = 10^{-5}$. Также для более стабильного процесса оптимизации приенялась техника *клипинга градиента* - ограничения значений градиента пороговыми значениями ($grad := max(grad, threshhold)$), обеспечивающий более плавную сходимость без *взрывов градиента*. Результат на валидационной выборке после 3 эпох файнтюнинга:

Метрика	Значение
Accuracy	0.99307
Precision	0.725280
Recall	0.85514
f1	0.78488

Figure 7: Изменение функции потерь в ходе дообучения



3.3 Интерпретация алгоритма

3.3.1 Значение Шэпли

Как было обозначенно в начале, алгоритмы требуется интерпретировать для последующей оценки реальными экспертами. Одним из основных методов является оценка влияния входных данных на результат. При помощи *SHAP (SHapley Additive exPlanation)* можно оценить вклад входных признаков на ответ модели. Несмотря на то, что *SHAP* предназначен для оценки вклада тгроков в каолиционной игре и вычисления основаны на расчете маржинального вклада игроков, при помощи интерпретации призников как игроков и вычисления маржинального вклада при помощи сэмплинга [13], эти значения будут являться хорошим критерием при анализе модели машинного обучения[14].

3.3.2 Анализ модели при помощи SHAP

Анализ BERT моделей с классификацией последовательности представленный в [15] подразумевает вычисления для каждого токена входной последовательности. Для применения подхода к текущей задаче восприятие разметки как бинарной классификации каждого члена последовательности так же полезно. В данном случае вычислен *SHAP* для пар элемент *i* - вероятность для элемента *j*.
В результате для предложения получилась матрица 8.

```
investigated_sample = X_test.iloc[0]["feature_text"]+" "+X_test.iloc[0]["pn_history"]
```

```
investigated_sample
```

```
'insomnia a 45 y/o f came to the clinic c/o nervousness. started gradually for the first time 3-4 weeks ago and since it started
it is getting progressively worse. describes it as anxiety over work achievements, constant throughout the day, relate to change
in her work from research to lectures, worsen before the lectures and no improvements factors. associated w/ decreased apetite,
headache and difficulty falling asleep. denies fever, chills, nausea, vomit, changes in weight, weakness, numbness, dizziness, p
alpitations, sweating or tremors. \r\n\r\nros: negative, except as above; pmhx: none; psh: none; all: nkda; meds: tylenol for he
adaches; fh: non relevant; sh: married. sexually active w/ husband. works as a english professor. no h/o smoking or illicit drug
use. etoh: occ.; caffeine: 5-6 cupps/day.'
```

Пользуясь такой матрицей можно по индексу в последовательности найти слова, ответственные за принятие решения сетью и понять логику получения ответа для какого-то региона.

Figure 8: Карта SHAP Values

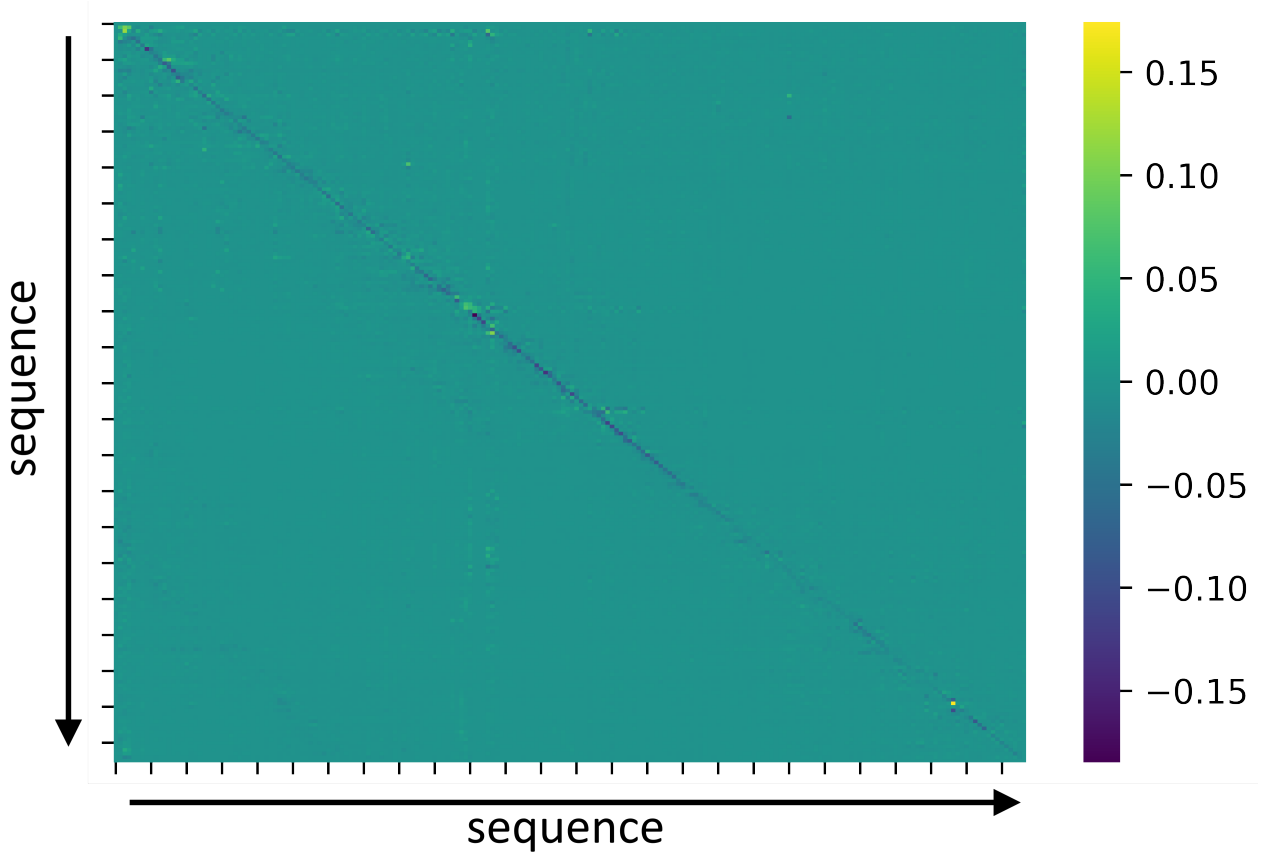
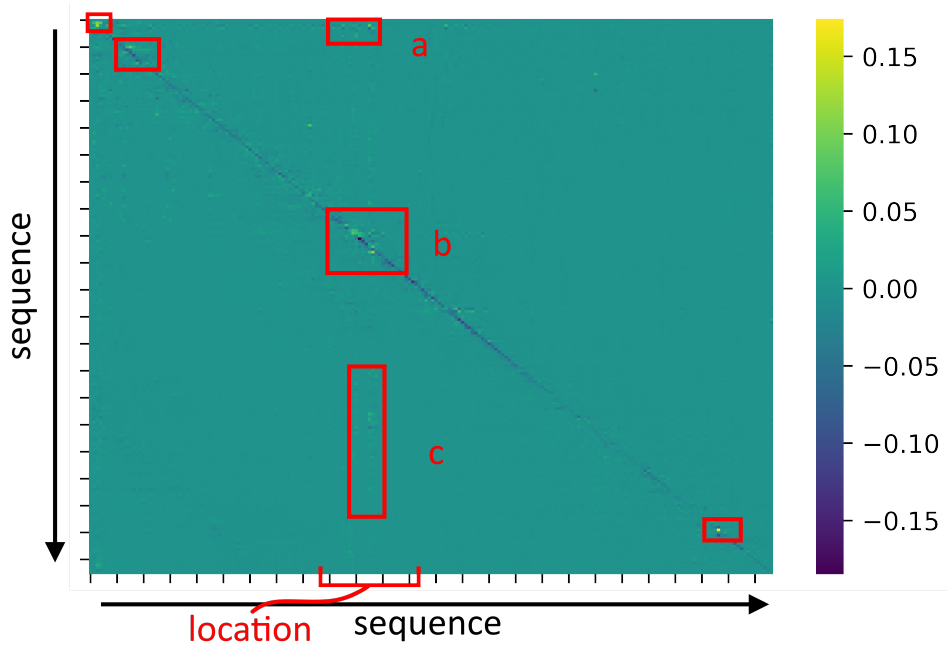


Figure 9: Карта SHAP Values с выделениями: красными рамками выделены токены с повышенными положительными SHAP; *a,b* и *c* соответствуют 1 найденному региону, при этом элементы в *b* наиболее близки к диагонали, то есть токены самого региона вложились в повышение вероятности, а *a* и *c* находятся ранее и после региона в последовательности, выступили в качестве контекста.



4 Заключение

В ходе работы были собраны основные статьи и технологии, отметившиеся в сфере решения задач обработки естественного языка. Подробно описана проблематика и специфика задач этой области формально и интуитивно отражены наиболее важные концепты, на которых основываются алгоритмы.

Была обозрена история развития механизма внимания и архитектур с его применением, приведены разъяснения о проектировании и главных идеях применения, разобраны основные свойства многомерных представлений данных в этих алгоритмах.

Технология BERT была применена к нестандартной задачи с медицинскими данными на естественном языке. Осуществлено сведение к стандартизированной задаче классификации, осуществлен процесс файнтюнинга претренированной модели.

Были применены фреймворки numpy, pandas, pytorch и hugging face transformers, являющиеся основными прикладными технологиями для решения подобных задач.

Применены SHAP Values для анализа работы решения, предложен метод интерпретации и визуализации для решенной задачи.

Список используемой литературы

[1] Tomas Mikolov, Kai Chen, Greg Corrado, Jeffrey Dean. “Efficient Estimation of Word Representations in Vector Space.” (2013).

[2] Piotr Bojanowski, Edouard Grave, Armand Joulin, Tomas Mikolov. “Enriching Word Vectors with Subword Information.” (2017).

[3] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N. Gomez, Lukasz Kaiser, Illia Polosukhin. “Attention Is All You Need.” (2017).

[4] Jacob Devlin, Ming-Wei Chang, Kenton Lee, Kristina Toutanova. “BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding.” (2018).

[5] Philip Gage. “A New Algorithm for Data Compression.” (1994).

[6] Geert Litjens, Thijs Kooi, Babak Ehteshami Bejnordi, Arnaud Arindra Adiyoso Setio, Francesco Ciompi, Mohsen Ghafoorian, Jeroen A.W.M. van der Laak, Bram van Ginneken, Clara I. Sanchez. “A Survey on Deep Learning in Medical Image Analysis.” (2017).

[7] Mike Schuster, Kaisuke Nakajima, ”Japanese and Korean voice search.” (2012)

[8] Kazemnejad Amirhossein, ”Transformer Architecture: The Positional Encoding.” (2019)

[9] Zachary C. Lipton, John Berkowitz, Charles Elkan ”A Critical Review of Recurrent Neural Networks for Sequence Learning.” (2015)

[10] Sepp Hochreiter, J¨urgen Schmidhuber ”Long short-term memory.” (1997)

[11] Dzmitry Bahdanau, Kyung Hyun Cho, Yoshua Bengio ”Neural Machine Translation by Jointly Learning to Align and Translate.” (2014)

[12] https://www.kaggle.com/competitions/nbme-score-clinical-patient-notes (2022)

[13] Scott M. Lundberg, Su-In Lee ”A Unified Approach to Interpreting Model Predictions.” (2017)

[14] Mukund Sundararajan, Amir Najmi ”The Many Shapley Values for Model Explanation.” (2020)

[15] Blaz Skrlj, Shane Sheehan, Nika Erzen, Marko Robnik-Sikonja, Saturnino Luz, Senja Pollak ”BERT meets Shapley: Extending SHAP Explanations to Transformer-based Classifiers.” (2020)