Детекция эмоций. Сравнение и анализ классических методов машинного обучения и методов обучения с трансформерами

A Preprint

Панин Никита Александрович
Факультет вычислительной математики и кибернетики
МГУ им. Ломоносова
s02200456@gse.cs.msu.ru

д.ф-м.н., профессор, Воронцов Константин Вячеславович
Факультет вычислительной математики и кибернетики
МГУ им. Ломоносова
vokov@forecsys.ru

Аннотация

В работе рассмотрена задача детекции эмоций на датасете, в основу которого вошел WASSA датасет из твитов для детекции эмоций. На выходе алгоритма классификации эмоций в твитах была одна из 5 эмоций: нейтральная эмоция, грусть, страх, радость, гнев. Были применены различные методы "классического"машинного обучения, такие как, SVM, логистическая регрессия, метод k-ближайших соседей и наивный байесовский классификатор. Также классификация эмоций была проведена с помощью файн-тюнинга нескольких версий BERT. Основной целью работы являлось проведение сравнительного анализа для классических моделей машинного обучения(wKNN, Multinomial Bayes Classifier, Logistic Regression, SVM) и для моделей глубокого обучения (в качестве предобученной модели брались BERT, RoBERTa, BERTweet и их large-версии). В результате исследования было показано, что по метрике accuracy для моделей классического обучения с tf-idf векторизацией текстов лучше всего работает SVM с RBF ядром ($accuracy \approx 0.8387$ на тесте), а наиболее качественные результаты получаются с помощью предложенной в исследовании модели с предобученным BERTweet ($accuracy \approx 0.88$ на тесте).

Ключевые слова Детекция эмоций · NLP

1 Введение

Детекция эмоций в текстах стала актуальным направлением исследований в области обработки естественного языка (NLP) и анализа тональности и привлекает все большее внимание в последние два десятилетия [27, 12]. Термины "распознавание эмоций"(emotion detection) и "анализ тональности"(sentiment analysis) часто используются как взаимозаменяемые, хотя между этими двумя понятиями существуют очевидные различия [4]. Анализ тональности в основном измеряет субъективное отношение с точки зрения полярности настроения: нейтральное, положительное, негативное. Выявление эмоций предполагает идентификацию более детальных эмоциональных состояний, например, счастье, гнев, страх, удивление.

Эмоции имеют множество применений в разных сферах. В маркетинге анализ предпочтений потребителей помогает улучшить бизнес-стратегии [5]. В социальных сетях распознавание агрессивных эмоций помогает выявить потенциальных преступников или террористов [7]. Мониторинг эмоций в реальном времени на основе данных социальных сетей может помочь в профилактике самоубийств [16].

Определение эмоций во время кризисов или катастроф позволяет понять чувства людей по отношению к конкретной ситуации, что способствует управлению в кризис и принятию важных решений [33].

1.1 Существующие решения

Ванг и др. [29] создали большой набор данных твитов, используя хэштеги эмоций. Они применили два различных классификатора, логистическую регрессию и Naïve Bayes, чтобы исследовать эффективность различных признаков, таких как n-граммы, лексикон эмоций и информация о части речи, для задачи идентификации эмоций. Наибольшая достигнутая точность составила 0,6557.

Мохаммад [22] создал корпус твитов, маркированных эмоциями, используя хэштеги. Он применил бинарные SVM, по одному для каждой из шести основных эмоций Экмана [15], и использовал наличие или отсутствие униграмм и биграмм в качестве бинарных признаков. Бинарные классификаторы смогли предсказать эмоции со сбалансированным F1-score 0,499.

Янссенс и др. [2] исследовали влияние использования слабых меток по сравнению с сильными метками на распознавание эмоций для корпуса, состоящего из 341 931 твита. Слабые метки были созданы путем использования хэштегов твитов, а сильные метки - с помощью краудсорсинга. Характеристики, извлеченные путем объединения n-грамм и TF-IDF (Term Frequency-Inverse Document Frequency), были применены к пяти алгоритмам классификации: Стохастический градиентный спуск, SVM, Naïve Bayes, Nearest Centroid и Ridge. Результаты показали снижение F1-score на 9,25% при использовании слабых меток.

К сожалению, классические методы машинного обучения не могут учесть последовательную природу текста, поэтому некоторые модели глубокого обучения, такие как рекуррентные нейронные сети (RNN), LSTM [1] и GRU [9], стали более перспективными в определении эмоций в тексте [21, 6, 3]. Хотя рекуррентные модели принимают во внимание последовательный характер текста и показывают передовые результаты для различных задач NLP, они обладают некоторыми слабостями: медленная скорость, необходимость обучения с нуля и ограниченная способность улавливать долгосрочные зависимости в тексте [26]. Также требуется большой объем размеченных данных для обучения. Подготовка большого объема размеченных данных является трудоемкой и дорогостоящей процедурой, и именно здесь вступает в игру перенос обучения(transfer learning). С его помощью можно добиться лучших результатов по сравнению с традиционными моделями глубокого обучения с гораздо меньшим количеством обучающей выборки. Предварительно обученные языковые модели, такие как BERT [13] (Bidirectional Encoder Representations from Transformers) и его варианты, OpenAI GPT (Generative Pre-trained Transformer) [25] и Transformer-XL [11], получили широкое распространение в различных задачах NLP и продемонстрировали впечатляющие результаты.

Некоторые работы используют предварительно обученные языковые модели для классификации эмоций или тональности в тексте. Например, исследование [24] использует BERT в качестве слоя эмбеддинга, после которого выводы передаются через слои CNN и BiLSTM для анализа настроений на бенгальском языке.

2 Постановка задачи

В данной работе была рассмотрена задача детекции эмоций на датасете, в основу которого вошел WASSA датасет из твитов для детекции эмоций [23]. Были применены различные методы "классического" машинного обучения, такие как, SVM, логистическая регрессия, метод k-ближайших соседей и наивный байесовский классификатор. Также классификация эмоций была проведена с помощью файн-тюнинга нескольких версий BERT.

Формально каждый твит d_i векторизовался в вектор $x_i \in \mathbb{R}^h$, описывающий структуру d_i, где h - размерность вектора x_i. Далее полученный эмбеддинг x_i поступал на вход одному из перечисленных выше алгоритмов классификации $\mathcal{A}$ и на выходе $\mathcal{A}(i)$ получали вероятности эмоций p_{ik}, $\sum_{k=1}^{5} p_{ik} = 1$. Итоговым ответом алгоритма была эмоция, соответсвующая наибольшей вероятности. В качестве эмоций рассматривалось 5 типов: нейтральная эмоция, грусть, страх, радость, гнев.

За основу бралась статья [19]. Основной целью работы являлось проведение сравнительного анализа полученных моделей. Также в работе предлагается сравнение новых моделей, таких как large-версии бертов, логистическая регрессия, wKNN, SVM с другими ядрами, с результатами из основной статьи.

3 Условия экспериментов

3.1 Описание данных

Данные[1], используемые для обучения и оценки эмоций в твитах, были частично получены из набора данных WASSA, представленного участникам на семинаре по компьютерным методам анализа субъективности, настроения и социальных сетей (WASSA-2017) [23]. Было выбрано по 1500 твитов для каждой из четырех эмоций: страх, грусть, радость и гнев, при этом информация об интесивности эмоций из датасета была исключена. К этим четырем категориям был добавлен еще один класс из 1500 нейтральных твитов, так как в исследовании[20] было показано, что наличие нейтрального класса важно для классификации, поскольку это позволяет модели не относить неизвестные эмоции к одному из изучаемых классов. Нейтральные твиты были взяты с CrowdFlower. Данные были разделены таким образом, чтобы обеспечить хороший баланс между классами [28]. Сбалансированный набор данных содержит одинаковое количество примеров для каждого класса, что гарантирует, что модель не будет уделять больше внимания крупным классам при классификации. Для обучения бралось 80% данных, для теста и валидации по 10%.

3.2 Метрики

Поскольку используемый датасет сбалансирован по классам, то в качестве основной метрики была взята доля правильных классификаций(accuracy), которая плохо работает в случае дисбаланса классов. Также сравнения велись по точности(Precision), полноте(Recall), F1 мере(F1) с макроусреднением.

Для каждого класса $y \in Y$:

- TP_y – верные положительные
- FP_y – ложные положительные
- FN_y – ложные отрицательные

Точность, полнота и F1 мера с макроусреднением:

$$Precision_{macro} = \frac{1}{|Y|} \sum_{y} \frac{TP_y}{TP_y + FP_y}$$

$$Recall_{macro} = \frac{1}{|Y|} \sum_{y} \frac{TP_y}{TP_y + FN_y}$$

$$F1_{macro} = \frac{2 * Precision_{macro} * Recall_{macro}}{Precision_{macro} + Recall_{macro}}$$

В дальнейшем изложении пометка "macro" будет опускаться.

3.3 Классическое машинное обучение

3.3.1 Предобработка текстов

"Сырые" твиты в большинстве своем это "грязные", сильно зашумленные данные. Это сопряжено прежде всего с природой твитов: люди часто пишут сокращениями, с ошибками и прочее. При предобработке текстов использовались стандартные методы в машинном обучении для подготовки текстовых данных на вход алгоритму:

- Эмотикон
  Поскольку через текст часто сложно передать эмоции, эмотиконы приобрели очень большую популярность и пользователи нередко используют их в своих твитах. В связи с этим эмотиконы были переведены в текстовый формат с помощью библиотеки Demoji Python, чтобы они помогали в идентификации эмоций в твитах [30].

[1]Данные были взяты с:
https://github.com/alexalbu98/Emotion-Detection-From-Tweets-Using-BERT-and-SVM-Ensemble-Model/blob/master/dataset.zip

- Хэштег
  Хэштеги - это фразы без пробелов с префиксом в виде символа решетки (#), которые часто используются пользователями для упоминания трендовой темы в Твиттере. Все хэштеги были заменены словами без символа решетки. Например, #hello заменяется на hello.
- Упоминание пользователей
  У каждого пользователя Твиттера есть имя пользователя, связанное с ним. Пользователи часто упоминают других пользователей в своих твитах через @username. В текстах все упоминания пользователей убирались.
- URL
  Пользователи часто делятся гиперссылками на другие веб-страницы в своих твитах. Какой-либо конкретный URL-адрес не важен для классификации текста и , если бы ссылки были оставлены, то это привело бы к очень разреженным признакам для текстов причем напрасно. Именно поэтому все URL-адреса в твитах были удалены.
- Фильтрация стоп-слов и стемминг
  Эмпирически доказано, что фильтрация стоп-слов в наборе данных повышает производительность и скорость вычислений [17], поэтому стоп-слова удалялись. Стемминг также использовался и это еще один метод повышения производительности модели путем сведения производных слов к грамматическому корню, называемому "stem".

3.3.2 Векторизация текстов

После предобработки тексты векторизовались с помощью наиболее популярного векторного представления – TF-IDF. Пусть d - документ из коллекции документов D, w_i - i-е слово из словаря W. Мощность (количество элементов) для множеств , как это принято в математике, будем обозначать $|\cdot|$. Тогда векторное представление каждого документа будет выглядеть так:

$$d_{vec} = \begin{pmatrix} tf-idf(w_1, d, D) \\ tf-idf(w_2, d, D) \\ \vdots \\ tf-idf(w_{|W|}, d, D) \end{pmatrix},$$

где

$$tf-idf(w_i, d, D) = \underbrace{tf(w_i, d)}_{term\ frequency} * \underbrace{idf(w_i, D)}_{inverse\ document\ frequency} =$$

$$= \underbrace{\frac{\sum_{w' \in d}[w_i = w']^2}{\sum_{w' \in d} 1}}_{tf(w_i, d)} * \underbrace{log\left(\frac{|D|}{\sum_{d \in D}[w_i \in d]}\right)}_{idf(w_i, D)}$$

Такие веса используются из предположения, что, чем больше документов, в которых встречается слово, тем меньше смысла оно несет в себе и следовательно его вес меньше.

3.3.3 Модели

1. wKNN
   Вместо классического KNN использовался взвешенный KNN (weighted KNN, wKNN) [14], в котором веса не равномерно распределены для ближайших соседей объекта, а обратно пропорционально от расстояний от объекта до ближайших соседей. Основными параметрами для этого метрического метода являются: расстояние между текстами и k ближайших соседей.

   В качестве метрики в метрических алгоритмах берут функции убывающие от "близости", поэтому будем отталкиваться именно от "близости" текстов. Согласно Хуанг [18], которая сравнивала различные меры близости для кластеризации текстов, лучшая оказалась косинусная мера:

$$cosine\_similarity(x, y) = \frac{x^T y}{\|x\|\|y\|},$$

[2]Используется нотация Айверсона

где x, y - векторные представления текстов, а $\|\cdot\|$ - норма вектора. В качестве косинусного расстояния бралась функция:

$$cosine\_distance(x, y) = 1 - cosine\_similarity(x, y)$$

При подборе k можно было воспользоваться эмпирическим правилом и брать $k = \frac{\sqrt{|D|}}{2} = 38$(в нашем случае), однако, поскольку это лишь эмпирическое правило, было решено попробовать перебрать k и оказалось, что лучшим выбором было $k = 52$

2. Logistic Regression
Логистическая регрессия - это алгоритм машинного обучения с учителем, используемый для задач бинарной или многоклассовой классификации. В контексте машинного обучения он используется для прогнозирования вероятности принадлежности объекта к определенному классу. Существует несколько подходов для использования logistic regression: либо использовать множество бинарных логистических классификаторов и на основе их выдавать ответ (one-vs-one или one-vs-rest) или изменить лосс функцию и на выходе вместо сигмоиды для предсказания вероятностей использовать софтмакс. В данной работе использовался второй подход. Формализуем задачу многоклассовой логистической регрессии.

   Линейный классификатор при произвольном числе классов $|Y|$

$$a(x) = \arg\max_{y \in Y} \langle w_y, x \rangle,\ x, w_y \in \mathbb{R},$$

   где $\langle \cdot, \cdot \rangle$ - скалярное произведение, w_y - веса для метки $y \in Y$.
   Вероятность того, что объект x относится к классу y:

$$P(y|x, w) = \frac{exp(\langle w_y, x \rangle)}{\sum_{z \in Y}(\langle w_z, x \rangle)} = \underset{y \in Y}{\text{SoftMax}}(\langle w_y, x \rangle)$$

   Обучение состоит в максимизации правдоподобия (log-loss) с регулизацией:

$$L(w) = \sum_{i=1}^{l} log(P(y_i|x_i, w) - \frac{\tau}{2} \sum_{y \in Y} \|w_y\|^2 \to \max_w$$

   Для τ было подобрано значение равное 1.

3. Multinomial Naive Bayes
Мультиномиальный байесовский классификатор (Multinomial Naive Bayes Classifier) является разновидностью наивного байесовского классификатора, который использует мультиномиальное распределение для моделирования данных. Этот алгоритм особенно хорошо подходит для решения задач классификации текста и анализа тональности, когда требуется разделить текстовые данные на несколько категорий. Несмотря на то, что классическая теория мультиномиального классификатора предполагает, что на вход подается векторизованные документы по частотам слов в каждом документе, существуют работы, в которых опытным путем было показано, что tf-idf дает лучше результаты [8]. По этой причине в работе был выбран tf-idf для мультиномиального байесовского классификатора.

4. Support Vector Classifier
Метод опорных векторов состоит в поиске оптимальной разделяющей гиперплоскости, которая приводит к максимизации ширины разделяющей полосы между классами, следовательно, к более уверенной классификации. SVC работает для бинарной классификации, поэтому в случае мультиклассовой классификации приходится строить несколько алгоритмов SVC и выбирать класс с помощью голосования. В качестве подхода построения таких алгоритмов была выбрана схема one-vs-one, поскольку при использовании one-vs-rest SVC придется обучать на большой выборке данных, что вычислительно менее эффективно, так как на больших данных SVC ресурсоемкий и будет долго обучаться.

3.4 Трансформеры

3.4.1 Предобработка и токенизация текстов

Для всех рассматриваемых версий BERT существуют свои специфичные токенизаторы. Перед подачей в токенизатор предложений[3] в каждом были убраны ссылки и имена пользователей, поскольку они не

[3]Под "предложением" будем понимать кусок текста, а не синтаксическое предложение

несут никакой информации об эмоциях. Несмотря на разницу в токенизаторах, отметим общие принципы:

- Нормализация, т.е. приведение к нормальному виду, что обычно делается путем приведения слов к нижнему регистру, контролирования специальных символов, в том числе, удаление пробельных символов.
- Токенизация, т.е. отображение слов в токены. Для BERT - WordPiece [32], для BERTweet и RoBERTa - byte-level BPE(Byte-Pair-Encoding) [31].
- Добавление специальных токенов, такик как '[CLS]', с помощью которого агрегируется информацию о предложении, используемая для задач классификации, а также '[PAD]', '[SEP]', '[EOS]'.

3.4.2 Модели

В работе рассмотрены три основные модели: BERT [13], RoBERTa [31], BERTweet [10], а также их large-аналоги, в которых больше слоев и больше обучаемых параметров. Для решения задачи классификации брался эмбеддинг, чья размерность 768, который соответствует позиции '[CLS]' токена в токенизированном предложении. Этот эмбеддинг проходил через dropout-слой с вероятностью обнуления элемента равной 0.3, чтобы снизить переобучение, далее через линейный слой и LogSoftMax, поскольку эта функция активации вычислительно более стабильна чем SoftMax (см. Рис. 1). Выбранной функцией

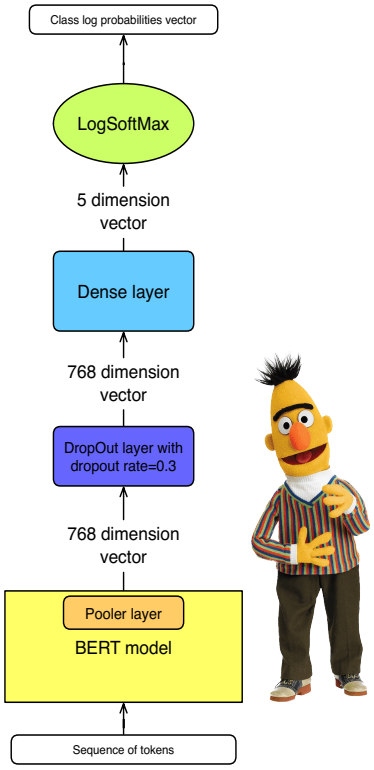

Рис. 1: Предложенная модель

потерь была NLLLoss(Negative Log Likelihood loss):

$$NLLLoss(y) = -log(y)$$

В качестве оптимизатора был выбран Adam. При обучении использовалось линейное уменьшение шага градиента и ограничение нормы градиента с максимальной нормой, равной 1, чтобы уменьшить вероятность появления исчезающего или взрывающегося градиента.

4 Эксперименты

4.1 Сравнение классических моделей машинного обучения.

Здесь в качестве лучшей модели из основной статьи для классического метода машинного обучения брался SVM с rbf-ядром, с ней, в частности, будет приведено сравнение с другими моделями (начертание далее — "SVM")

4.1.1 Дизайн эксперимента.

Модели для сравнения: взвешенный метод ближайших соседей(wKNN), логистическая регрессия(LR), мультиномиальный наивный байесовский классификатор(MNB), метод опорных векторов(SVM)

Гиперпараметры для моделей:

1. wKNN:
   - количество соседей - 52
   - расстояние - косинусное
2. MNB:
   - сглаживание по Лапласу (Laplace smoothing)
3. LR:
   - регуляризация - L2-регуляризация с параметром регуляризации равным 1
4. SVM
   - ядро - ядра выбирались разные, но лучший результат(см. таблицу 1) показало $RBF(x_i, x_j) = \exp\left(-\gamma\|x_i - x_j\|^2\right)$ с $\gamma = \frac{1}{n\_features * X.var()}$, где $n\_features$ - количество признаков, $X.var()$ - дисперсия, получающаяся, если вытянуть таблицу объекты-признаки в один массив и считать ее у получившихся значений.
   - регуляризация - L2-регуляризация с параметром регуляризации равным 1

Ядро	precision	recall	F1-score
$Linear$	0.852	0.849	0.849
$Polynomial\ (degree = 3)$	0.777	0.7	0.708
RBF	0.873	0.871	0.871

Таблица 1: Различные ядра SVM на валидации.

4.1.2 Результаты и выводы

	precision	recall	F1-score
$wKNN$	0.701	0.691	0.693
MNB	0.778	0.78	0.776
LR	0.834	0.832	0.833
SVM	0.873	0.871	0.871

Таблица 2: Сравнение метрик качества моделей на валидации.

Посмотрим на таблицу 2, на которой изображены метрики качества моделей на валидационной выборке. Видно, что хуже всех с классификацией эмоций справляется wKNN, возможно, это связанно с тем, что для представления текстов в виде векторов необходимо использовать векторы большой размерности(размер словаря), а это неизбежно для метрических алгоритмов приводит к "проклятию

размерности", из-за чего wKNN становится не эффективным. Лучший результат был получен с помощью SVM с rbf-ядром благодаря способности данного метода находить нелинейные зависимости сложной структруры, поэтому ни wKNN, ни логистическая регрессия, ни мультиномиальный байесовский классификатор не смогли улучшить качество классификации по сравнению с моделью из основной статьи.

Для того, чтобы лучше понять как модель SVM работает при детекции каждой эмоции была построена матрица ошибок на тестовой выборке, которую можно увидеть на рис. 2.

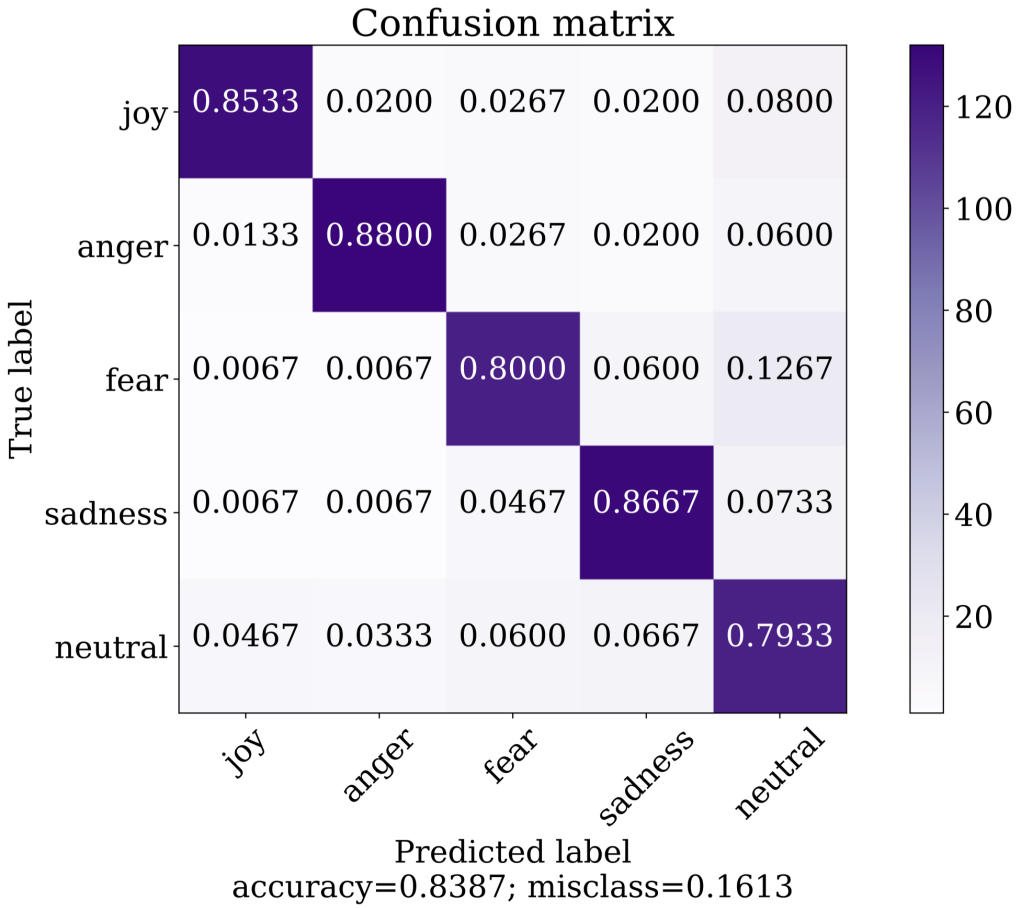

Рис. 2: Предложенная модель

	Joy	Anger	Fear	Sadness	Neutral
Precision	0.92	0.93	0.83	0.84	0.70
Recall	0.85	0.88	0.80	0.87	0.79
F1-score	0.89	0.90	0.82	0.85	0.74

Таблица 3: Precision, recall, F1-score для SVM на тестовой выборке

Результаты показывают, что модель лучше всего работает на объектах класса "гнев" и хуже всего на "нейтральной" эмоции. Это может быть из-за того, что нейтральный класс был взят из другого распределения[4] (другого датасета) и таких объектов меньше чем объектов других классов, взятых из одного датасета.

[4]Напоминание: нейтральный класс был добавлен из CrowdFlower, остальные из WASSA

4.2 Трансформеры. Влияние эпох.

4.2.1 Дизайн эксперимента

- Модель: Предложенная модель с BERT
- Эпохи: 5, 10
- lr[5] $= 2 * 10^{-5}$ с линейным убыванием lr на каждом шаге оптимизации

4.2.2 Результаты и выводы

Модель с BERT очень мощная и уже предобучена, поэтому, как правило, не требуется много эпох для файн-тюнинга, что и сделало ее столь популярной. Посмотрим как разное количество эпох влияет на обучение.

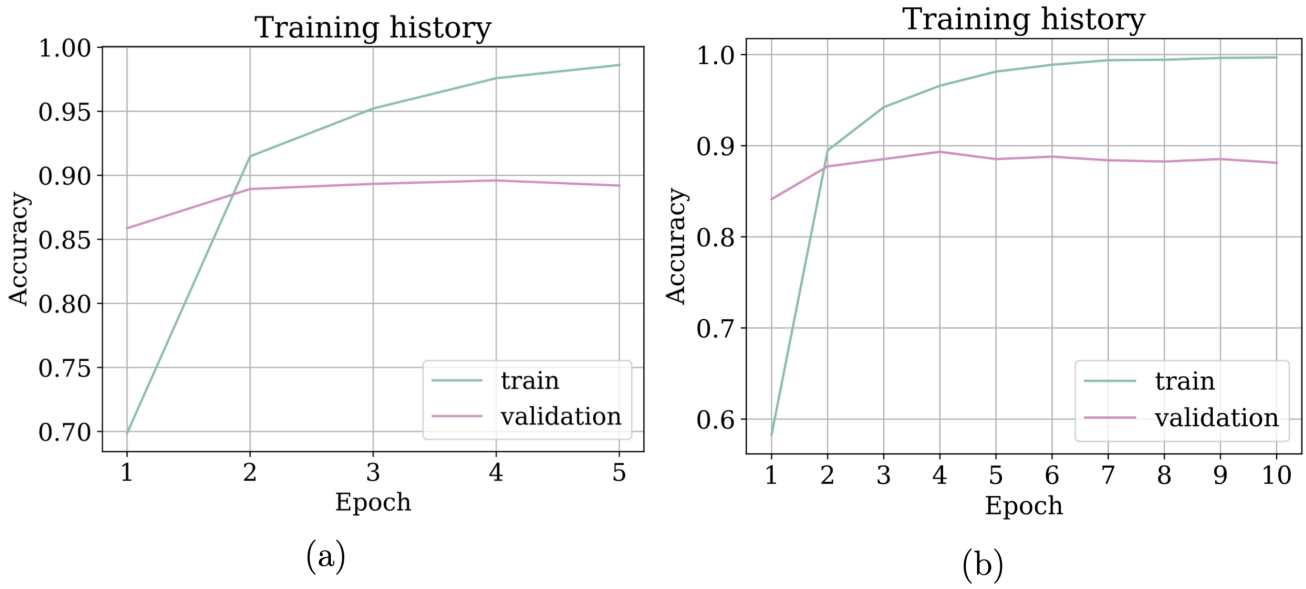

Рис. 3

Видно, что модель уже на небольшом количестве эпох выходит на плато на валидационной выборке и дальнейшее увеличение количества эпох бессмысленно, поэтому далее для небольших версий BERT использовалось 5 эпох.

4.3 Трансформеры. Влияние размера батча.

4.3.1 Дизайн эксперимента

Все параметры, не перечисленные далее, такие же как и в предыдущем эксперименте

- Размеры батча: 8, 16, 32, 64

4.3.2 Результаты и выводы

Посмотрим как сильно влияет размер батча на предсказания: Не сложно заметить, что влияние размера батча оказывается очень маленьким и доля верных предсказанных ответов изменяется в пределах 0.88 до 0.905 при том, что увеличение размера батча приводит к более длительному обучению модели. Несмотря на то, что 16 объектов в батче дает неплохие результаты и по времени, и по метрике, в дальнейшем в экспериментах будет использоваться размер батча равный 32.

[5] lr - learning rate(темп обучения)

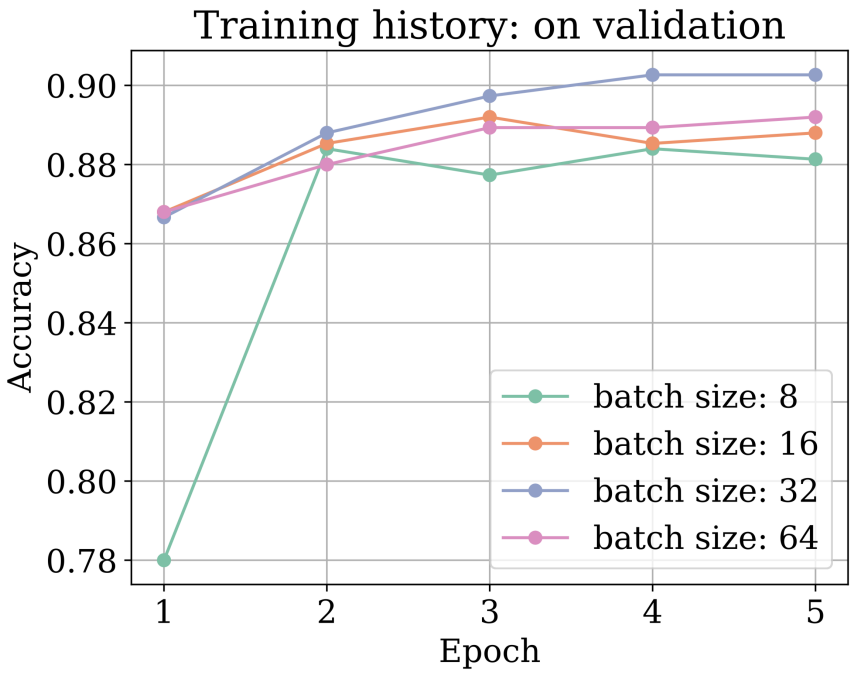

Рис. 4

4.4 Трансформеры. Темп обучения.

4.4.1 Дизайн эксперимента

В качестве базовой модели по-прежнему бралась предложенная модель с BERT, 5 эпох обучения и размер батча равный 32. Для lr бралось 6 значений по логарифмической сетке от 10^{-6} до 10^{-3} и дальше на каждом шаге оптимизации lr линейно уменьшался.

4.4.2 Результаты и выводы

Видно, что из-за слишком маленького количества эпох при малых значениях модель не дообучается ($lr = 10^{-6}$, $lr = 3.126 * 10^{-6}$, $lr = 10^{-4}$), а при слишком больших ($lr = 10^{-3}$) вообще не способна обучаться, поскольку градиент становится большим и, видимо, перескакивает точку минимума. Оптимально брать значения в диапазоне от $2 * 10^{-5}$ до 10^{-5}. Для версий BERT с малым количеством параметров в дальнейшем будет браться $lr = 2 * 10^{-5}$.

4.5 Сравнение различных версий BERT

Здесь в качестве лучшей модели из основной статьи для бертов брался BERTweet, с ним, в частности, будет приведено сравнение с другими версиями берта (начертание далее — "BERTweet")

4.5.1 Дизайн эксперимента

В качестве моделей рассматривались BERT, BERT-large, RoBERTa, RoBERTa-large, BERTweet, BERTweet-large. Для моделей с маленьким числом обучаемых параметров(BERT, RoBERTa, BERTweet) хорошо сработали те параметры, что рассматривались в предыдущих экспериментах. Для моделей с суффиксом "large", у которых больше параметров, пришлось подбирать новые гиперпараметры, потому что модели не дообучались. Эти гиперпараметры оказались одинаковыми для каждой large-версии:

- Количество эпох: 7
- Размер батча: 32
- Темп обучения(lr): $5 * 10^{-6}$ с линейным убыванием на каждом шаге оптимизации.

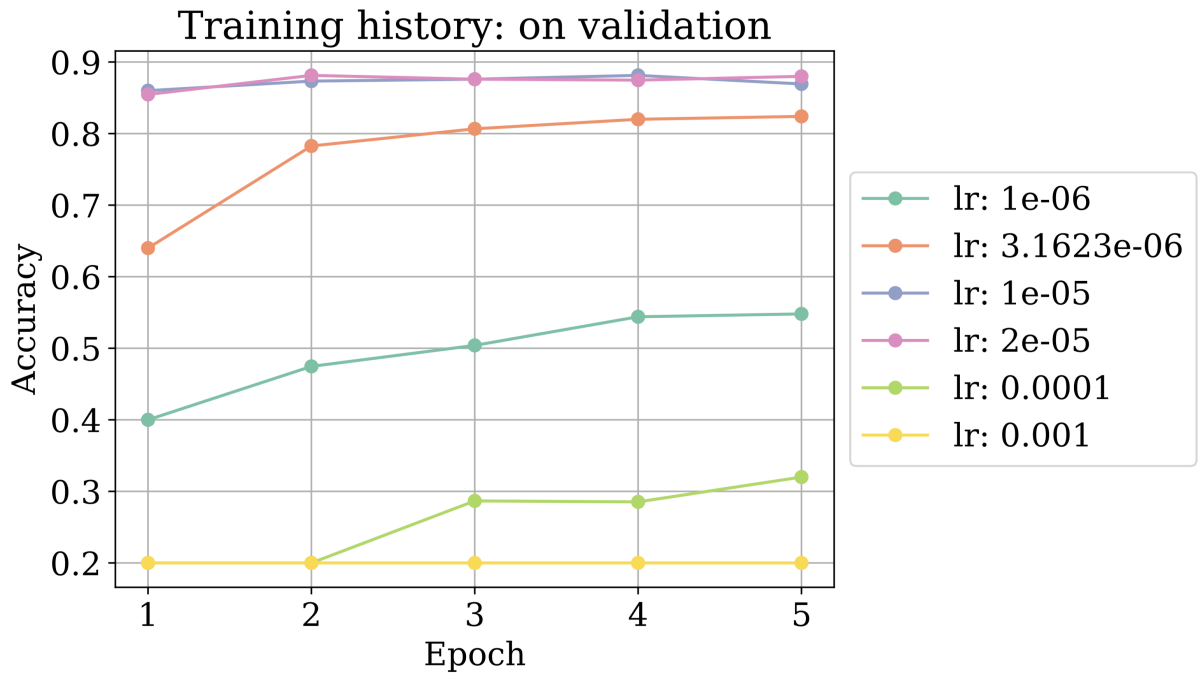

Рис. 5

4.5.2 Результаты и выводы

После обучения моделей на датасете из твитов, описанном в начале работы, были построены графики доли правильных ответов от эпох(см. Рис. 6)

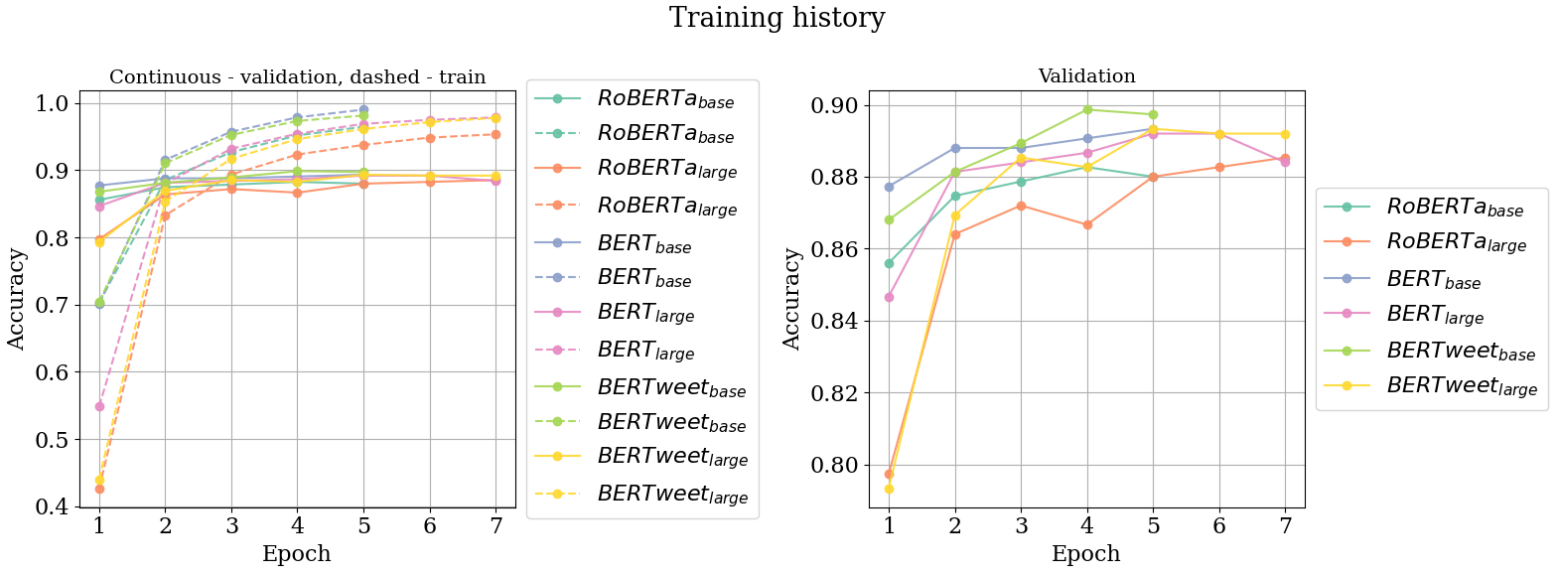

Рис. 6

На левом графике видно, что все модели, в целом, имеют небольшое переобучение, а значит являются потенциально достаточно надежными моделями, то есть на других данных из твитов будут давать примерно такой же результат как и на валидации.

	precision	recall	F1-score
$RoBERTa_{base}$	0.8827	0.8834	0.8829
$RoBERTa_{large}$	0.8861	0.8853	0.8855
$BERT_{base}$	0.8968	0.8933	0.8940
$BERT_{large}$	0.8923	0.8920	0.8921
$BERTweet_{base}$	0.9004	0.8987	0.8992
$BERTweet_{large}$	0.8954	0.8933	0.8939

Таблица 4: Сравнение метрик качества моделей на валидации.

Несмотря на то, что большее количество параметров может приводить к более высоким результатам по метрикам качества, очевидно из правого графика, что единственная модель, которой удалось побить качество своей маленькой версии является $RoBERTa_{large}$.

Из всех моделей самое высокое значение $accuracy = 0.8987$ было получено у модели $BERTweet$ из статьи, что объясняется двумя факторами. Во-первых, $BERTweet$ была предобучена по той же схеме, что и $RoBERTa$, а она, в свою очередь, является улучшенной версией $BERT$, которая превзошла $BERT$ во многих задачах. Во-вторых, $BERTweet$ предобучена на 850M твитов и как раз ориентирована на наш датасет.

Таким образом, предположение о том, что large-версии бертов смогут заметно улушить качество исходной модели из статьи, оказалось в данной задаче безуспешным.

Также была построена матрица ошибок(см. Рис. 7). На ней видно, что accuracy модели в сравнении с SVM значительно улучшилось на таких классах как: 'радость' (на $\approx$0.06), 'страх' (на $\approx$0.03), 'грусть' (на $\approx$0.04) и 'нейтральная' эмоция (на $\approx$0.11). Тем не менее есть одна эмоция, а именно, 'гнев', которая лучше предсказывается SVM. (на $\approx$0.035). Заметим еще, что в случае использования предложенной модели на основе BERTweet, получилось добиться accuracy по всем классам на $\approx$0.04 лучше чем у SVM.

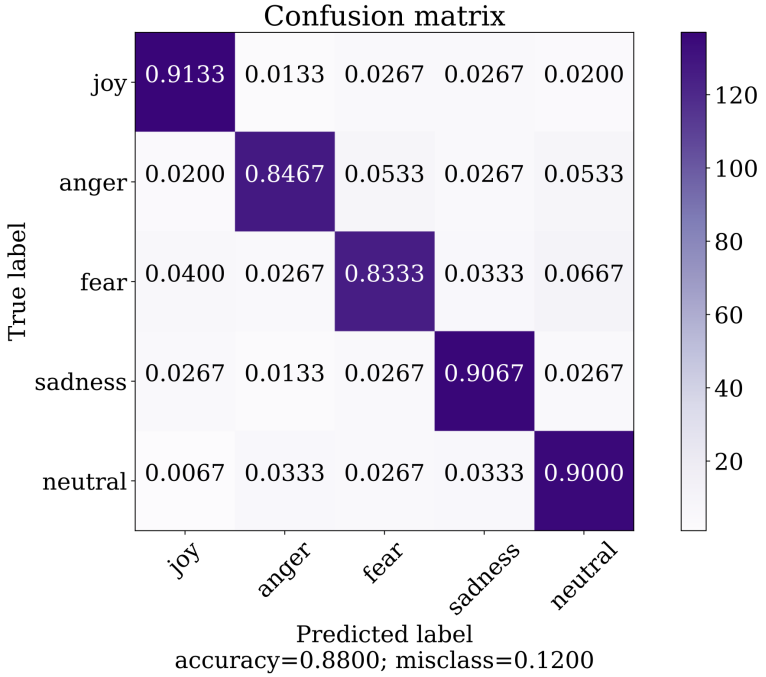

Рис. 7: Матрица ошибок для BERTweet на тестовой выборке

5 Заключение

В данной работе были проанализированы и сравнены несколько моделей для детекции эмоций. Проведены эксперименты на данных, состоящих из твитов, рассмотрены некоторые элементы предобработки таких данных. Из исследования можно сделать следующие выводы:

- wKNN, логистическая регрессия, модели SVM с линейным, полиномиальным ядрами, мультиномиальный байесовский классификатор хуже сработали чем модель SVM из исходной статьи.
- SVM с rbf-ядром и TF-IDF при правильной обработке данных дает неплохие результаты ($accuracy = 0.8387$) при том, что обучение занимает меньше времени чем у трансформеров.
- Модель с BERTweet позволяет получить наилучшее качество для данных из твитов с $accuracy = 0.88$.
- BERT версии с большим количеством параметров, так называемые, large-модели, не показали в данной задаче заметного улучшения в сравнении с base-версиями (меньшим числом параметров) и не смогли превзойти BERTweet из исходной статьи.
- BERT версии не требуют большого количества эпох для обучения, что связанно с предобученностью моделей
- Размер батча слабо влиял на качество, важно было подобрать правильный темп обучения. В итоге для малых версий бертов подошел $lr = 2 * 10^{-5}$, а для больших $lr = 5 * 10^{-6}$

В дальнейшем исследовании, можно попробовать использовать более умные эмбеддинги для классических моделей машинного обучения, которые не просто собирают статистики, а могут запоминать контекст слов (GloVe, FastText, ELMO), также было бы интересно проверить гипотезу о возможном улучшении классификации в случае добавления в эмбеддинги информации о наиболее часто встречаемой эмоции для этого слова, собранной из словарей, в которых каждому слову сопоставлена эмоция.

Список литературы

[1] Hochreiter S, Schmidhuber J. Long short-term memory. Neural Comput 9(8), pages 1735–1780, 1997.

[2] O. Janssens, S. Verstockt, E. Mannens, S. Van Hoecke, and R. Van de Walle. Influence of weak labels for emotion recognition of tweets. In In Prasath R., O'Reilly P., Kathirvalavakumar T. (Eds.), Mining Intelligence and Knowledge Exploration, Springer, pages 108–118, 2014.

[3] Xu G, Li W, Liu J . A social emotion classification approach using multi-model fusion. In Future Generat Comput Syst 102, pages 347–356, 2020.

[4] A. Yadollahi, A. G. Shahraki, and O. R. Zaiane. Current state of text sentiment analysis from opinion to emotion mining. In ACM Comput. Surv., vol. 50, no. 2, page 1–33, Jan. 2017.

[5] E. Cambria. Affective computing and sentiment analysis. In IEEE Intell. Syst., vol. 31, no. 2, pages 102–107, Mar./Apr. 2016.

[6] Chatterjee A, Gupta U, Chinnakotla MK, Srikanth R, Galley M, Agrawal P. Understanding emotions in text using deep learning and big data. In Comput Human Behav 93, page 309–317, 2019.

[7] M. Cheong and V. C. S. Lee. A microblogging-based approach to terrorism informatics: Exploration and chronicling civilian sentiment and response to terrorism events via twitter. In Information Systems Frontiers, vol. 13, no. 1, Mar. 2011.

[8] Chingmuankim and Prof.Rajni Jindal. A comparative study of naive bayes classifierswith improved techniqueon text classification.

[9] Cho K, Van Merriënboer B, Gulcehre C, Bahdanau D, Bougares F, Schwenk H, Bengio Y . Learning phrase representations using rnn encoder-decoder for statistical machine translation. In Conference on Empirical Methods in Natural Language Processingf. EMNLP, 2014.

[10] D. Q. Nguyen, T. Vu, and A. T. Nguyen. Bertweet: A pre-trained language model for english tweets. In In Proc. of the 2020 Conference on Empirical Methods in Natural Language Processing: System Demonstrations, pages 9–14, Nov. 2020.

[11] Dai Z, Yang Z, Yang Y, Carbonell J, Le QV, Salakhutdinov R . Transformer-xl: Attentive language models beyond a fixed-length context. In Associat Comput Linguist, 2019.

[12] Jiawen Deng and Fuji Ren. A survey of textual emotion recognition and its challenges. 2021.

[13] Devlin J, Chang M, Lee K, Toutanova K . Bert: pre-training of deep bidirectional transformers for language understanding. In Association for Computational Linguistics, 2018.

[14] Sahibsingh A Dudani. The distance-weighted k-nearest-neighbor rule. In IEEE Transactions on Systems, Man, and Cybernetics, pages (4):325–327, 1976.

[15] P. Ekman. An argument for basic emotions, cognition and emotion. pages 169–200, 1992.

[16] F. Ren, X. Kang, and C. Quan. Examining accumulated emoional traits in suicide blogs with an emotion topic model. In IEEE J. Biomed. Health Inform., vol. 20, no. 5, Sep. 2016.

[17] H. Saif, M. Fernandez, Y. He, and H. Alani. On Stopwords, Filtering and Data Sparsity for Sentiment Analysis of Twitter. In In Proc. of the Ninth International Conference on Language Resources and Evaluation (LREC'14), Reykjavik, Iceland, pages 810–817, 2014.

[18] Anna Huang. Similarity measures for text document clustering. In In Proceedings of the sixth new zealand computer science research student conference (NZCSRSC2008), Christchurch, New Zealand, pages volume 4, pages 9–56, 2008.

[19] Ionut-Alexandru ALBU, Stelian SPINU. Emotion detection from tweets using a bert and svm ensemble model. 2022.

[20] M. Koppel and J. Schler. The importance of neutral examples for learning sentiment. In Computational Intelligence, vol. 22, no. 2, pages 100–109, 2006.

[21] Kratzwald B, Ilić S, Kraus M, Feuerriegel S, Prendinger H. Deep learning for affective computing: Text-based emotion recognition in decision support. In Decision Support Syst 115, page 24–35, 2018.

[22] S. M. Mohammad. Emotional tweets. In Proc. of the First Joint Conference on Lexical and Computational Semantics - Volume 1: Proceedings of the Main Conference and the Shared Task, and Volume 2: Proceedings of the Sixth International Workshop on Semantic Evaluation, pages 246–255, 2012.

[23] S. M. Mohammad and F. Bravo-Marquez. Wassa-2017 shared task on emotion intensity. In Proc. of the 8th Workshop on Computational Approaches to Subjectivity, Sentiment and Social Media Analysis, Copenhagen, Denmark, pages 34–49, 2017.

[24] Prottasha NJ, Sami AA, Kowsher M, Murad SA, Bairagi AK, Masud M, Baz M . Transfer learning for sentiment analysis using bert based supervised fine-tuning. 2022.

[25] Radford A, Narasimhan K, Salimans T, Sutskever I. Improving language understanding by generative pre-training. 2018.

[26] Hochreiter S. The vanishing gradient problem during learning recurrent neural nets and problem solutions. In Int J Uncertainty, Fuzziness Knowl-Based Syst 6(02), pages 107–116, 1998.

[27] C. Strapparava and R. Mihalcea. Affect detection in texts 13. In The Oxford Handbook of Affective Computing, page 184–216, 2015.

[28] T. Borovicka, M. Jirina Jr., P. Kordik, and M. Jirina. Selecting representative data sets. In In Karahoca A. (Ed.), Advances in Data Mining Knowledge Discovery and Applications, IntechOpen, pages 43–70, 2012.

[29] W. Wang, L. Chen, K. Thirunarayan, and A. P. Sheth. Harnessing twitter 'big data' for automatic emotion identification. In Proc. of the 2012 International Conference on Privacy, Security, Risk and Trust and 2012 International Conference on Social Computing, pages 587–592, 2012.

[30] W. Wolny. Emotion analysis of twitter data that use emoticons and emoji ideograms. In Information Systems Development: Complexity in Information Systems Development (ISD2016 Proceedings), Katowice, Poland, pages 476–483, 2016.

[31] Yinhan Liu, Myle Ott, Naman Goyal, Jingfei Du, Mandar Joshi, Danqi Chen, Omer Levy, Mike Lewis, Luke Zettlemoyer†, Veselin Stoyanov. Roberta: A robustly optimized bert pretraining approach.

[32] Yonghui Wu, Mike Schuster, Zhifeng Chen, Quoc V. Le, Mohammad Norouzi. Google's neural machine translation system: Bridging the gap between human and machine translation. 2016.

[33] Z. Ahmad, R. Jindal, A. Ekbal, and P. Bhattachharyya. Borrow from rich cousin: Transfer learning for emotion detection using cross lingual embedding. In Expert Syst. Appl., vol. 139, Jan. 2020.