«Московский физико-технический институт (национальный исследовательский университет)»
Физтех-школа прикладной математики и информатики
Кафедра «Интеллектуальные системы»

Пилькевич Антон Александрович

Оптимизация критерия, заданного нейросетевой моделью, в задаче детоксификации текста

03.03.01 – Прикладные математика и физика

Выпускная квалификационная работа бакалавра

Научный руководитель:
д.ф.-м.н. Стрижов Вадим Викторович
Консультант:
Попов Артём Сергеевич

Москва
2022

Содержание

1 Введение **4**

2 Постановка задачи **7**

2.1 Детоксификация, как машинный перевод 7

2.2 Детоксификация, как текстовая стилизация 9

2.3 Проблема отсутствия дифференцируемости 10

3 Обзор литературы **12**

3.1 Задача стилизации 12

3.2 Автоматическая оценка качества стилизации 12

3.3 Текстовые порождающие модели 13

4 Адаптер **14**

4.1 Адаптер — аппроксимация векторного представления . . . 14

4.2 Обучение адаптера 16

4.3 Использование адаптера в дообучении детоксификатора . 17

5 Вычислительный эксперимент **19**

5.1 Данные 19

5.2 Метрики 19

5.3 Baseline T5 20

5.4 Одновременное обучение адаптера и детоксификатора . . . 20

5.5 Поочередное обучение адаптера и детоксификатора 21

6 Заключение **24**

Аннотация

Один из способов оценки качества в задачах генерации текста — использование предобученной (например, на задачу классификации) глубокой нейросетевой модели. Например, модель, решающая задачу определения токсичности текста, может использоваться для оценки качества задачи детоксификации текста. Логично предположить, что оценивающую модель следует использовать для задания функции потерь, чтобы улучшить качество решения. Нейросетевые модели для обработки текста работают с ограниченным словарём входных токенов, и у разных моделей словарь может отличаться. В силу отсутствия инъективного отображения между токенами моделей, наивный способ использования оценивающей модели поверх результатов основной невозможен. Данная работа предлагает способ решения отсутствия дифференцируемости такой функции потерь на примере задачи детоксификации текстов. Предлагается обучить новый входной эмбеддинг-слой оценочной модели, называемый адаптером, который будет принимать распределения вероятностей токенов, выданных моделью детоксификатором. В работе предложен способ обучения адаптера с использованием в качестве функции потерь дивергенции Кульбака-Лейблера между истинной вероятностью токсичности и вероятностью, полученной при использовании адаптера. Также предложен способ одновременного обучения детоксификатора и адаптера, приводящие к улучшению качества модели. Проведенные эксперименты показывают эффективность предложенного метода

1 Введение

На сегодняшний день для решения задач в области обработки естественного языка активно используется подход переноса обучения (transfer learning). Глубокие нейросетевые модели на основе архитектуры трансформер [1, 2] предобучаются на большом корпусе текстов, а затем дообучаются на целевую задачу (не обязательно совпадающую с задачей предобучения).

В последнее время начинает набирать популярность использование предобученных моделей в качестве метрик качества в задачах генерации текста. Далее мы будем называть такие модели *оценивающими*, а модели решающие основную задачу — *целевыми*. Сгенерированный целевой моделью текст подаётся на вход оценивающей модели, решающей определённую задачу, а её результат используется в качестве значения метрики качества. Например, в задаче текстовой стилизации в качестве оценочной модели может использоваться модель, оценивающая естественность полученного текста на основе XLM-R [3, 4].

Возникает естественный вопрос: можно ли использовать предобученную модель не только для оценивания качества целевой модели, но и для её обучения? Возможно ли сконструировать функцию потерь, на основе весов оценивающей модели, улучшающую качество решения целевой задачи? В данной работе рассматриваются эти вопросы на примере задачи детоксификации текста.

Детоксификация текста - это задача текстовой стилизации, где требуется по токсичному предложению построить его нейтральный вариант. Под токсичностью в большинстве случаев подразумевается наличие обсценной лексики. Задача детоксификации можно поставить как задачу машинного перевода(sequence-to-sequence), где на вход модели подаётся токсичный текст, а на выходе генерируется нейтральный вариант. Основная архитектура для решения таких задач - энкодер-декодер на основе трансформера [2]. Кодировщик трансформера принимает на вход последовательность BPE-токенов (символьные n-граммы) из ограниченного множества, называемого словарём, и кодирует их в последовательность

векторов. Декодировщик трансформера, используя выходы кодировщика, генерирует токены по очереди в авторегрессионной манере.

Наивный способ использования оценивающей модели для задания функции потерь — подать выходы декодировщика на вход этой модели, а затем сопоставить выходное значение оценивающей модели с значением для истинного результирующего предложения. Например, если модель оценивает токсичность предложения по шкале от нуля до единицы, выход модели должен быть близок к нулю.

Такой подход имеет огромный минус — если между токенами словарей моделей нет инъективного отображения, получившаяся композиция моделей не будет дифференцируема, и обучение градиентными методами будет невозможно. На практике инъективность отображения встречается только при полном совпадении словарей, например, если до начала обучения задать словарь целевой модели равным словарю оценивающей модели. Проблема такого подхода в том, что это делает невозможным использование предобученных моделей (если у них не совпадает словарь с оценивающей моделью) для инициализации весов целевой модели, что может серьёзно сказаться на итоговом качестве.

Схожая проблема, связанная с потерей дифференцируемости, возникает в задачах генерации текстов при помощи порождающих моделей [5,6]. Дабы решить эту проблему одни подходы используют идеи обучения с подкреплением [7], основанные на аппроксимации градиентов и методе Монте-Карло. Другие подходы используют идею gumbel-softmax для апроксимации one-hot векторов полученных предложений [8]. Но основная проблема дифференцируемости в этих работах связана с операцией $\arg\max$, в то время как в данной работе всё сводится к различным словарям детоксификатора и оценочной модели. Единственный подход, который можно попробовать использовать, — идея обучения с подкрепление, но в описанных работах он показывает плохие результаты и труден в обучении.

В данной работе описан способ использования оценивающей модели в качестве функции потерь, не накладывающий требования на словарь целевой модели. Предлагается во время обучения заменить вход-

ной эмбеддинг-слой оценивающей модели на линейный слой, называемый *адаптером*. Адаптер принимает на вход распределения вероятностей токенов, выданных моделью-детоксификатором, и возвращает аппроксимацию входных эмбеддингов оценочной модели. Для обучения адаптера используется функция потерь в виде дивергенции Кульбака-Лейблера, которая принимает вероятность токсичности для истинного предложения при использовании оценочной модели и вероятность токсичности детоксифицированного при использовании оценочной модели с адаптером. Кроме того данная работа описывает алгоритм обучения адаптера и модели-детоксификатора. Благодаря предложенному методу удалось добиться прироста метрики, заданной оценочной моделью, и увеличить целевую метрики, представляющую собой произведение метрики оценочной модели и chrF1 [9].

2 Постановка задачи

2.1 Детоксификация, как машинный перевод

В данной работе рассматривается задача *детоксификации* текста. Требуется по токсичному предложение построить нейтральный без потери семантического смысла. Под токсичным предложением может подразумеваться как предложение с обсценной лексикой, так и предложения оскорбительного характера без её употребления.

На этапе обучения доступен корпус из токсичных предложений, для каждого из которых известен нейтральный вариант. Примеры приведены в таблице 1. Поэтому задачу детоксификации удобно рассматривать, как задачу машинного перевода (seq-to-seq).

Данные представляют собой параллельный корпус текстов (s_t, s_d), где s_t — токсичное предложение из датасета, s_d — его нейтральная версия. Обозначим два пространства с различными стилями: T — пространство токсичных предложений, D — пространство нейтральных предложений. Тогда требуется построить отображение из T в D.

Для работы нейросетевых моделей с текстом требуется разбить его на токены. В данной работе для разбиения текста на токены используется BPE-токенизатор [10].

Пускай имеется *словарь* V_f, содержащий все возможные токены для модели-детоксификатора. Тогда τ_f — *токенизатор*, такой что

$$\tau_f : T \cup D \to (V_f)^n,$$

где n — фиксированная длина последовательности токенов. Следует заметить, что в данной постановке задачи токенизатор τ_f и словарь V_f не дообучаются под задачу детоксификации, а являются фиксированными. Значения τ_f и V_f зависят только от предобученной модели, использующейся для инициализации весов целевой модели.

Пусть также имеется два предложения $s_t \in T$ и $s_d \in D$. Тогда $t = \tau_f(s_t), d = \tau_f(s_d)$ — их разбиения на токены. Введём *детокисфикатор*

f_θ, такую что

$$f_\theta : (V_f)^n \to [0,1]^{n \times |V_f|},$$
$$f_\theta(*|d_{<i}, t) \in [0,1]^{|V_f|},$$
$$\sum_{d_i \in V_f} f_\theta(d_i|d_{<i}, t) = 1,$$

где $f_\theta(*|d_{<i}, t)$ — распределение вероятностей i-го токена при условии токсичного предложения и сгенерированных $i-1$ токенах. В данной работе детоксификатор имеет архитектуру кодировщик-декодировщик, трансформер T5 [2]. Как отмечалось ранее, параметры детоксификатора будут инициализироваться параметрами модели предобученной на другую задачу. Мотивация такой инициализации заключается в том, что предобученная модель обучалась на корпусе текстов в разы больше и у неё сформировалось «понимание» языка. Подход дообучения модели хорошо себя зарекомендовал, и позволяет качественно решать поставленную задачу при наличии маленького корпуса обучающих данных [11]. Более детально о выбранной модели будет описано в секции 5.3.

Детоксификатор, как модель кодировщик-декодировщик, генерирует токены в авторегрессионном подходе. Токены генерируются один за одним, на каждом шаге выбирая токен с наибольшей вероятностью при условии предыдущих:

$$\arg\max_{d_i \in V_f} f_\theta(d_i|d_{<i}, t).$$

Заметим, что после генерации токена, означающего конец предложения, оставшиеся токены будут не информативными и заполняются специальным токеном. После генерации последовательности, она передаётся в функцию *детокенизации* Text_f, которая составляет текст s_{detox} на основе последовательности токенов и словаря детоксификатора V_f:

$$s_{\text{detox}} = \text{Text}_f\left(\{\arg\max_{d_i} f_\theta(d_i|d_{<i}, t)\}_{i=1}^n\right).$$

В простейшей постановке задачи машинного перевода имеется функция потерь задаётся кросс-энтропией:

$$\mathcal{L}_{\text{CE}} = -\sum_{i=1}^{n} \log f_\theta(d_i | d_{<i}, t).$$

Тогда решаемая задача записывается как:

$$\mathcal{L}_{\text{CE}} \longrightarrow \min_\theta .$$

2.2 Детоксификация, как текстовая стилизация

Отметим, что в задаче машинного перевода кросс-энтропийная функция потерь явно не учитывает нейтральность полученных предложений, то есть не учитывается задача стилизации. Это распространённая проблема в задачах переноса текстового стиля при наличии параллельного корпуса текстов [12]. Один из возможных способ учитывать нейтральность полученного предложения — это подсчёт потенциально токсичных и оскорбительных слов. При таком подходе возникает много трудностей. Требуется учитывать различную морфологию, семантику, контекст, а также иметь в наличии список потенциально токсичных и оскорбительных слов. Поэтому, учитывая успехи нейросетевых моделей в задачах классификации [13,14], наиболее эффективным подходом оценки степени токсичности являются нейросети.

Обозначим нейросетевую модель, отвечающую за оценку токсичности предложения, как g_{toxic}. Далее будем называть её *оценочной моделью*. Как и модели-детоксификатора, оценочная модель имеет фиксированный словарь V_g и фиксированный обученый BPE-токенизатор $\tau_g : T \cup D \to (V_g)^n$, необязательно совпадающие с V_f, τ_f. С учётом приведённых выше обозначений и шкалы токсичности от нуля до одного можем записать:

$$g_{\text{toxic}} : (V_g)^n \to [0, 1].$$

Будем рассматривать выход модели, как вероятность токсичности предложения. Введём детоксифицированное предложение s_{detox}, полученное моделью-детоксификатором для входного токсичного предложения s_t, как это было описано в предыдущей секции 2.1.

Тогда решение следующей задачи будет явно учитывать задачу переноса стиля:

$$\mathcal{L}_{\text{TP}}(s_{\text{detox}}) = g_{\text{toxic}}\big(\tau_g(s_{\text{detox}})\big) \longrightarrow \min_{\theta}.$$

2.3 Проблема отсутствия дифференцируемости

Как было описано в секции 2.1, для лучшего решения задачи берётся инициализация параметров детоксификатора предобученной моделью. Это приводит к тому, что токенизатор τ_f и словарь V_f являются фиксированными, так как иначе не будет эффекта от инициализации. В качестве оценочной модели будет использоваться обученная модель, которая также имеет фиксированный токенизатор τ_g и словарь V_g, которые скорее всего отличны от τ_f, V_f.

При использовании оценочной модели g_{toxic} в качестве функции потерь возникает проблема отсутствия дифференцируемости по параметрам детоксификатора θ. Дифференцируемость нарушается при построении s_{detox} из-за операции $\arg\max$, а также по причине различных токенизаторов у детоксификатора и оценочной модели. Вторая проблема в первую очередь возникает из-за различных словарей V_f, V_g. Нет гарантий, что существует инъективное отображение между токенами τ_f и τ_g. Проще всего увидеть проблему на примере:

$$\textit{ты слишком токсичный} \longrightarrow \begin{cases} [\textit{ты},\ \textit{слишком},\ \textit{токс},\ \textit{ич},\ \textit{ный}], & \text{от } \tau_f, \\ [\textit{ты},\ \#UNK,\ \textit{токсичн},\ \textit{ый}], & \text{от } \tau_g. \end{cases}$$

В случае одинаковых словарей проблема отображения между токенами решилась автоматически, так как используются одинаковые токены. А проблему не дифференцируемости операции $\arg\max$ можно решить, например, при использовании gumbel-softmax или других подхо-

дов. Для случая наличия инъективного отображения между словарём V_f и V_g всё сведётся к дифференцируемости $\arg\max$, так как мы просто не будем использовать часть токенов V_g, которые нам не нужны.

В последующей главе будет предложен метод позволяющий обойти проблему дифференцируемости $\arg\max$ и различия словарей V_f, V_g.

3 Обзор литературы

3.1 Задача стилизации

Как было описано ранее, задачу текстовой стилизации для параллельного корпуса текстов можно рассматривать, как задачу машинного перевода. Данная работа в качестве базового решение задачи детоксификации основывается на таком подходе, дообучая трансформер T5 [2]. Но при таком подходе никак не учитывается корректность пар в датасете, точное соответствие заданному стилю и насколько расплывчаты «границы» заданного стиля. Поэтому важной составляющей задачи стилизации является факт, что семантический смысл любого текста может быть выражен в любом стиле [12].

3.2 Автоматическая оценка качества стилизации

Одной из составляющих качества задачи стилизации и машинного перевода является оценка сохранения смысла. Для случая параллельных корпусов текстов хорошо себя показывает chrF [9]. Он зачастую демонстрирует высокую корреляцию с человеческой оценкой качества сохранения смысла [4, 15]. Но данная метрика никак не учитывает стиль получаемых предложений. В качестве метрик, отвечающих за соответствие стилю, хорошо себя показывают нейросетевые подходы [4]. Но при этом для случая детоксификации русскоязычных текстов такой подход слабо коррелирует с человеческими оценками [15].

ChrF — F-score на основе символьных n-грамм. Введём chrP — доля символьных n-грамм из предлагаемого предложения, которые имеются в оригинальном. И chrR — доля символьных n-грамм из оригинального предложения, которые представлены в предлагаемом. Тогда финальная формула:

$$\text{chrF}_\beta = (1 + \beta^2)\frac{\text{chrP} \cdot \text{chrR}}{\beta^2\text{chrP} + \text{chrR}},$$

где chrP, chrR считаются как среднее по всем предложениям.

В качестве метрики качества стилизации будем рассматривать **Style transfer accuracy (STA)** — вероятность токсичности предложения. Метрика задаётся нейросетевой оценочной моделью g_{toxic}. В качестве модели g_{toxic} используется предобученный Conversational RuBERT[1], дообученный на задачу определения токсичности[2].

3.3 Текстовые порождающие модели

Схожая проблема «потери» дифференцируемости встречается у текстовых порождающих моделях. Проблема заключается в том, что дискриминатор должен работать с последовательностью токенов на входе. Но чтобы получить последовательность токенов необходимо применить операцию $\arg\max$ для выхода генератора, что приводит к отсутствию дифференцируемости. Дабы решить эту проблему одни подходы предлагают использовать обучение с подкреплением [7], gumbel-softmax [8] и другие «ухищрения» [5, 6]. При этом отличительная особенность проблемы дифференцируемости в данной работы заключается в том, что параметры, словарь токенов V_g и токенизатор τ_g оценочной модели g_{toxic} фиксированы, а также зафиксирован словарь V_f и токенизатор τ_g детоксификатора f_θ. Наиболее простым подходом для реализации в данной работы является RL-подход, но зачастую он показывает плохие результаты для порождающих моделей и сложен в обучении. Поэтому был предложен подход с использованием адаптера, который будет описан в следующей главе.

[1] `https://huggingface.co/DeepPavlov/rubert-base-cased-conversational`

[2] `https://huggingface.co/SkolkovoInstitute/russian_toxicity_classifier`

4 Адаптер

4.1 Адаптер — аппроксимация векторного представления

Как отмечалось в секции 2.3, главная проблема при использовании оценочной модели g_{toxic} в качестве функции потерь является отсутствие дифференцируемости по параметрам детоксификатора θ. Предлагается избавиться от явной генерации нейтрального предложения s_{detox} детоксификатором для оценки степень токсичности получившегося предложения.

Введём произвольное предложение $s \in T \cup D$ и операцию $\textbf{one-hot}_g$ такую, что

$$\begin{gathered}\textbf{one-hot}_g : V_g \to \{0,1\}^{|V_g|}, \\ \left(\textbf{one-hot}_g(i)\right)_{j=i} = 1, \\ \left(\textbf{one-hot}_g(i)\right)_{j\neq i} = 0.\end{gathered}$$

Также введём матрицу входных эмбеддингов оценочной модели E_g. *Эмбеддингом токена* называется его векторное представление, причём $E_g \in \mathbb{R}^{|V_g|\times e_g}$, где e_g — размер входных эмбеддингов оценочной модели.

Оценочная модель g_{toxic} принимает на вход токены $\tau_g(s)$ сопоставляя им эмбеддинги из матрицы эмбеддингов E_g. Эту операцию можно записать следующим образом:

$$\begin{gathered}O_g E_g \in \mathbb{R}^{n\times e_g}, \\ O_g \in \{0,1\}^{n\times|V_g|},\end{gathered}$$

где i-я строчка в матрице O_g есть one-hot вектор для i-го токена $\tau_g(s)$.

Выход детоксификатора $f_\theta\big(\tau_f(s)\big) \in [0,1]^{n\times|V_f|}$, как было описано в секции 2.1, есть распределение вероятностей токенов из V_f. Введём *адаптер* $A \in \mathbb{R}^{|V_f|\times e_g}$, апроксимирущий входные эмбеддинги оценочной

модели следующим образом:

$$f_\theta\big(\tau_f(s)\big)A \approx O_g E_g. \qquad (\star)$$

При фиксированных параметрах оценочной модели это эквиваленто записи:

$$g^*_{\text{toxic}}\big(f_\theta\big(\tau_f(s)\big)\big) \approx g_{\text{toxic}}\big(\tau_g(s)\big),$$
$$g^*_{\text{toxic}} : [0,1]^{n\times|V_f|} \to [0,1],$$

где g^*_{toxic} — оценочная модель g_{toxic}, в которой заменили входную матрицу эмбеддингов E_g на адаптер A. Иллюстрация схемы работы изображена на рис. 1.

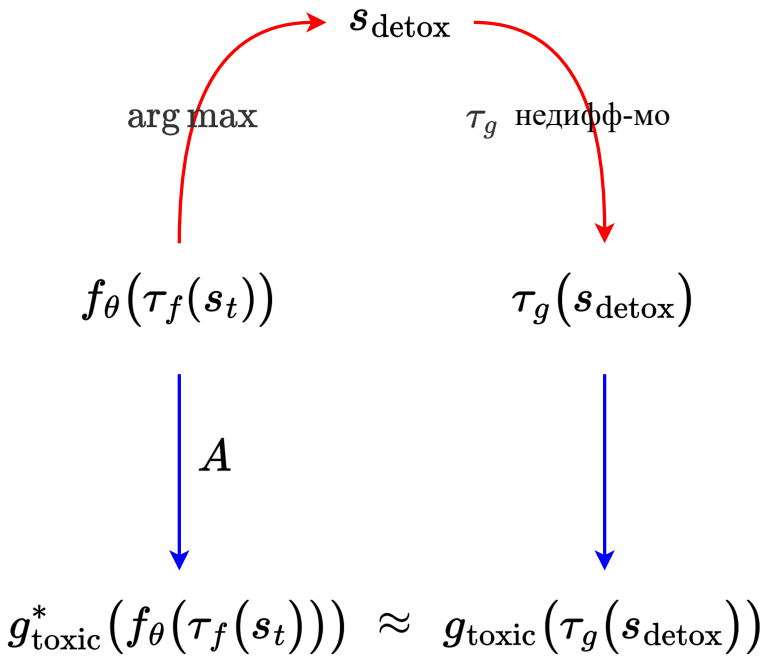

Рис. 1: Схема работы адаптера и решение проблемы дифференцируемости. Синими стрелками изображены операции не нарушающие дифференцируемость, красными - нарушающие.

В таком подходе можно интерпретировать выходное распределение детоксификатора $f_\theta\big(\tau_f(s)\big)$, как "зашумлённые"one-hot вектора. Так как матрица эмбеддингов до этого принимала вектора со значениеями в $\{0,1\}$, а теперь принимает со вектора значениемями на непрерывном

отрезке $[0, 1]$. По сути в таком подходе строится взвешенная сумма эмбеддингов всех токенов словаря V_f.

4.2 Обучение адаптера

Пусть обучаемые параметры — это только параметры адаптера A, все остальные параметры g^*_{toxic} фиксированы, как и параметры g_{toxic}. Также будем считать, что имеется обученная модель кодировщик-декодировщик на задачу детоксификации f_θ, как это было описано в секции 2.1.

Пусть имеется пара предложений s_t, s_d — токсичное и его нейтральный вариант соответственно. Введём две вероятности токсичности:

$$P_{\text{toxic}} = g_{\text{toxic}}\big(\tau_g(s_d)\big),$$
$$P^*_{\text{toxic}} = g^*_{\text{toxic}}\big(f_\theta\big(\tau_g(s_t)\big)\big).$$

Как было описано в предыдущей секции, мы требуем от адаптера выполнения аппроксимации $(\star)$. Но явно задавать функцию потерь для выполнения этого свойства не разумно, так как мы хотим использовать выход оценочной модели, а не только выход эмбеддинг слоя. Поэтому требуется выполнения аппроксимации на выход моделей: $P_{\text{toxic}} \approx P^*_{\text{toxic}}$.

Дабы достичь требуемого свойства аппроксимации определим функцию потерь для обучения адаптера, как дивергенцию Кульбака-Лейблера:

$$D_{\text{KL}}\left(P_{\text{toxic}} \parallel P^*_{\text{toxic}}\right) = P_{\text{toxic}} \cdot \log\left(\frac{P_{\text{toxic}}}{P^*_{\text{toxic}}}\right) + (1 - P_{\text{toxic}}) \cdot \log\left(\frac{1 - P_{\text{toxic}}}{1 - P^*_{\text{toxic}}}\right).$$

Тогда при обучении адаптера требуется решить следующую задачу:

$$D_{\text{KL}}\left(P_{\text{toxic}} \parallel P^*_{\text{toxic}}\right) \longrightarrow \min_A.$$

Причём важно отметить, что все параметры оценочной модели и модели-детоксификатора являются фиксированными. Схема обучения адаптера изображена на рис. 2.

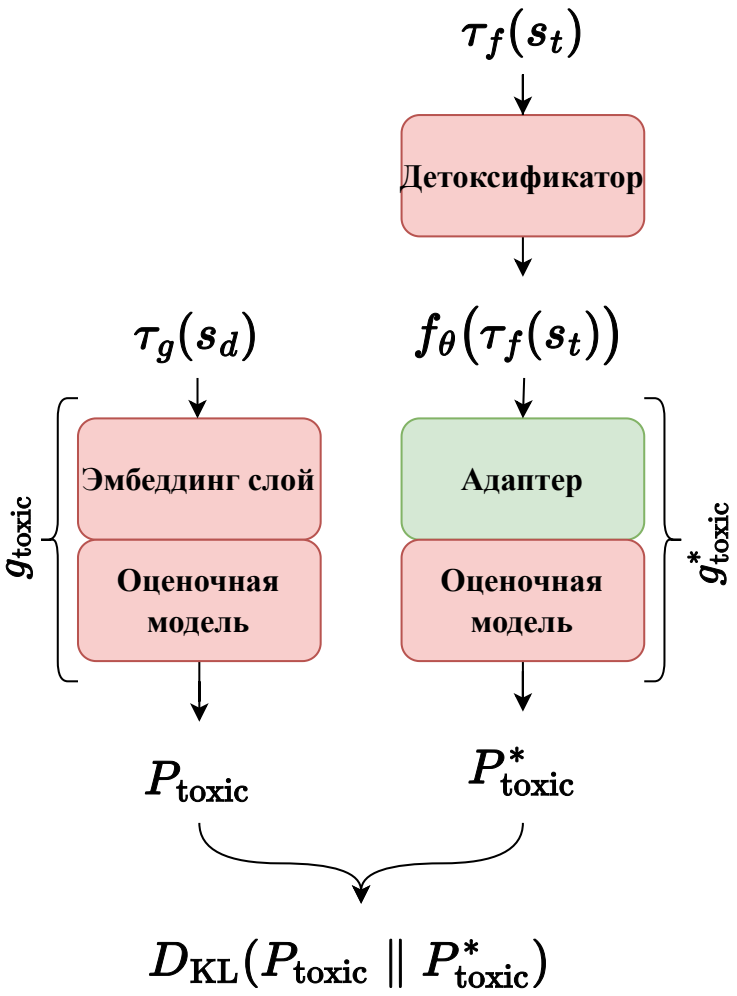

Рис. 2: Схема обучения адаптера. Красными блоками изображены фиксированные параметры моделей. Зелёным — обучаемые параметры. В данном случае только параметры адаптера.

4.3 Использование адаптера в дообучении детоксификатора

В этой секции будем считать, что имеется обученный детоксификатор f_θ на задачу машинного перевода и адаптер A для модели g^*_{toxic}. Отметим, что адаптер был обучен для фиксированных параметров детоксифакатора. Поэтому при дообучении детоксификатора будет инвалидироваться результат обучения адаптера, так как адаптер обучался для других параметров детоксификатора. Дабы избежать этого, требуется модифицировать алгоритм дообучения детоксификатора. Основная идея предложенной модификации основана на обучении порождающих моделей [16].

Введём следующие обозначения:

$$\text{CE} = \mathcal{L}_{\text{CE}},$$
$$\text{TP} = P^*_{\text{toxic}}.$$

Введём *агрегирующую функцию потерь* $\mathcal{F} : \mathbb{R}^2 \to \mathbb{R}$, которая будет принимать значения кросс-энтропии $\mathcal{L}_{\text{CE}}$ и вероятности токсичности P^*_{toxic}. Тогда предложенный алгоритм можно описать следующим образом:

1. N батчей обучается детоксиификатор f_θ, минимизируя функцию потерь $\mathcal{F}(\text{CE}, \text{TP})$, при фиксированных параметрах адаптера A.

2. M батчей обучается адаптер A, минимизируя D_{KL}, как описано в секции 4.2, при фиксированных параметрах детоксификатора f_θ.

5 Вычислительный эксперимент

5.1 Данные

Обучающие данные и данные для оценивания качества взяты из соревнования *«Russian Text Detoxification Based on Parallel Corpora»* [17]. Предложения были собраны с таких платформ, как «Одноклассники»[3], «Pikabu»[4], «Twitter»[5]. Как было описано раннее, датасет представляет себе параллельный корпус русскоязычных предложений, состоящий из токсичного варианта и его перефразированного нейтрального варианта. Нейтральная версия токсичного предложения была получена при помощи толокерской разметки[6]. У каждого токсичного предложения имеется от одного до трёх нейтральных вариантов.

Размер обучающего датасета составляет $N_{\text{train}} = 11136$ пар. Из которых 10% использовались для валидации во время обучения. Размер тестового датасета составляет $N_{\text{test}} = 800$ предложений. Примеры:

Токсичное предложение	Нейтральный вариант
«Её муженька козла на кол надо посадить.»	*«Её мужа нужно наказать.»*
«Это твари а не люди»	*«Это плохие люди.»*
«Да такой же урод и выдал.»	*«Этот же человек и посоветовал.»*
«А тем более с долбоебами.»	*«А тем более с ними.»*
«Надо расстреливать тех, кто говорит про пандемию.»	*«Надо наказывать тех, кто говорит про пандемию.»*

Таблица 1: Примеры пар предложений представленных в датасете.

5.2 Метрики

В качестве метрик для оценки качества детоксификации использовались chrF [9], Style transfer accuracy (STA), которые были описаны в

[3] https://www.kaggle.com/datasets/alexandersemiletov/toxic-russian-comments

[4] https://www.kaggle.com/datasets/blackmoon/russian-language-toxic-comments

[5] http://study.mokoron.com/

[6] https://toloka.yandex.ru/

секции 3.2. Тем самым chrF1 отвечает за сохранения смысла по сравнению с истинным нейтральным предложением, а STA отвечает за перенос стиля. В качестве целевой метрики используется произведение chrF1 и STA, дабы учесть влияние каждой из задач.

5.3 Baseline T5

В качестве исходной точки для оценки эффекта предложенных подходов используется подход решения задачи детоксификации, как задачи машинного перевода, как это было описанно в секции 2.1.

В качестве детоксификатора f_θ используется трансформер T5 [2], то есть архитектура кодировщик-декодировщик. Так как размер датасета в данной работе мал, то наиболее выгодным решением является использование предобученную архитектуры и дообучение её на задачу детоксификации. В качестве предобученной модели берётся ruT5-base[7], обученная на задаче seq-to-seq при испольховании датасет размером 300Gb.

5.4 Одновременное обучение адаптера и детоксификатора

В этой и следующей секции в качестве исходной модели детоксификатора f_θ используется модель полученная в предыдущей секции 5.3, после дообучения на задачу детоксификации, как на задачу машинного перевода.

Изначальная идея метода заключалась в одновременном обучении адаптера A и модели детоксификатора f_θ. Причём адаптер на вход принимал выходные эмбеддинги модели детоксификатора. Решаемую в это случае задачу можно записать через агрегирующую функцию потерь $\mathcal{F}$ следующим образом:

$$\mathcal{F}(\text{CE},\, \text{TP}) \longrightarrow \min_{\theta, A}.$$

[7] https://huggingface.co/sberbank-ai/ruT5-base

Но такой подход естественным образом приводит к переобучению адаптера. Адаптеру выгодно возвращать "шум"для оценочной модели g_{toxic^*}, чтобы уменьшить значения стилизационной функции потерь $\mathcal{L}_{\text{TP}}$. Как видно из таблицы 2, данный подход приводил к переобучению. Целевая метрика STA*chrF1 ухудшалась на 8%, что сильнее всего связано с уменьшением STA-метрики на 10%.

Следующая идея заключалась в использование softmax от выходных логитов модели детоксификатора f_θ в качестве входов адаптера при обучении на одинаковую функцию потерь $\mathcal{F}$. В этом случае выход детоксификатора можно интерпретировать, как one-hot вектора детоксификатора, как это было описано в секции 4.1. Но также, как и в предыдущем эксперименте, такой подход обучения детоксификатора приводит к переобучению модели, что видно из результатов в таблице 2. При этом результат получается лучше, чем при использовании эмбеддингов, что связано с меньшей "свободой"для адаптера и возможностей для переобучения.

В обоих подходов в качестве агрегирующей функции потерь $\mathcal{F}$ использовалось произведение: $\mathcal{F}(\text{CE}, \text{TP}) = \text{CE} \cdot \text{TP}$. Данный выбор был обусловлен тем, что целевой метрикой является STA*ChrF1.

Подход	$\mathcal{F}$	STA	chrF1	STA*chrF1
seq-to-seq	CE	0.742	0.577	0.428
Adapter on embs	$\text{CE} \cdot \text{TP}$	0.639	0.544	0.348
Adapter on logits	$\text{CE} \cdot \text{TP}$	0.708	0.569	0.403

Таблица 2: Результаты подхода, когда детоксификатор и адаптер оптимизируют общую функцию потерь $\mathcal{F}$.

5.5 Поочередное обучение адаптера и детоксификатора

Следующий подход заключался в раздельном обучении модели детоксификатора и адаптера при использовании различных функций потерь. Сперва обучался адаптер A также, как это было описано в секции 4.2. Но далее, в отличии от финальной версии предложенного метода,

на одном и том же батче сперва обучался детоксификатор при фиксированном адаптаре на $\mathcal{F}$, а потом дообучался адаптер при фиксированном детоксификаторе на D_{KL}. Далее такой подход будем называть как *Same batch*.

В качестве функции потерь $\mathcal{F}(\mathrm{CE}, \mathrm{TP})$ использовались два варианта:

$$\mathcal{F}(\mathrm{CE}, \mathrm{TP}) = \mathrm{CE} \cdot \mathrm{TP},$$
$$\mathcal{F}(\mathrm{CE}, \mathrm{TP}) = w_1 \cdot \mathrm{CE} + w_2 \cdot \mathrm{TP}.$$

Наилучшим образом себя показали коэффициенты для взвешенной функции потерь: $w_1 = 1, w_2 = 10$. Это объясняется тем, что модель детоксификатор до этого уже была обучена на кросс-энтропию и важно обращать внимание на функцию потерь, отвечающую за стилизацию, и взвешенная функция потерь позволяет сделать это явно. В данной постановке обучения проводились три запуска каждого из экспериментов с различными seed. Как видно из таблицы 3, есть значимое улучшение STA на 3%, но при этом происходит уменьшение chrF1. В результате чего целевая метрика в виде их произведение почти не изменяется. При этом использование различных функций потерь $\mathcal{F}$ даёт практически одинаковый результат.

Подход	$\mathcal{F}$	STA	chrF1	STA*chrF1
seq-to-seq	CE	0.744 ± 0.010	0.575 ± 0.002	0.430 ± 0.042
Same batch	$\mathrm{CE} \cdot \mathrm{TP}$	0.776 ± 0.015	0.569 ± 0.004	0.442 ± 0.006
Same batch	$\mathrm{CE} + 10 \cdot \mathrm{TP}$	0.774 ± 0.011	0.561 ± 0.012	0.435 ± 0.013

Таблица 3: Эксперименты по проверке статистической значимости подхода с обучением детоксификатора и адаптера на одном батче при использовании разных функций потерь, Same batch.

После этого использовалось финальный подход обучения, предложенная в секции 4.3. Будет называть его *GAN style*. Также как и в предыдущем эксперименте использовались два различных варианта функции

потерь $\mathcal{F}$:

$$\mathcal{F}(\text{CE}, \text{TP}) = \text{CE} \cdot \text{TP},$$
$$\mathcal{F}(\text{CE}, \text{TP}) = w_1 \cdot \text{CE} + w_2 \cdot \text{TP}.$$

При таком подходе обучения наилучшим образом себя показал вариант с взвешенной функцией потерь и явным предпочтением для стилизационной функции потерь: $w_1 = 1, w_2 = 10$. Результаты отображены в таблице 4. Удалось достичь улучшения STA на 7.4% и STA*chrF1 на 3.5%.

Также проводился эксперимент с дообучение детоксификатора только на стилизационную функцию потерь TP. При таком подходе происходит явное переобучение, что видно из метрик: STA ≈ 1.0 и ChrF1 в 5 раз меньше. Если посмотреть на сгенерированные предложения, то окажется, что детоксификатор выдаёт одно и тоже предложение для любого входа.

Подход	$\mathcal{F}$	STA	chrF1	STA*chrF1
seq-to-seq	CE	0.739	0.578	0.427
GAN style	TP	0.998	0.119	0.119
GAN style	$\text{CE} \cdot \text{TP}$	0.754	0.574	0.439
GAN style	$\text{CE} + 10 \cdot \text{TP}$	**0.813**	0.569	**0.462**

Таблица 4: Результаты экспериментов по оценки подхода обучения на различные агрегирующие функции в стиле порождающих моделей. То есть поочередное обучения детоксификатора на одних батчах, адаптера — на других.

6 Заключение

Данная работа предлагает метод использования обученных нейросетевых моделей в качестве функции потерь в задачах обработки естественного языка. Предложенный метод решает проблему отсутствия дифференцируемости такой функции потерь, возникающей из-за различных словарей токенизаторов оценочной и целевой модели, которая может инициализироваться предобученной моделью. Описанная проблема и предложенный метод рассматривались для задачи детоксификации текстов. Предложенный подход доказывает свою работоспособность и эффективность на основе результатов проведенных экспериментов. Также опубликован код для воспроизведение финального подхода[8]

В последующих работах необходимо обобщить предложенный метод на другие оценочные модели. Также требуется проверить работоспособность метода в других задачах. Отдельный интерес представляет сравнение с методами из текстовых порождающих моделей в текущей постановке задачи, так и сравнение предложенного метода для задачи генерации текстов против текущих подходов порождающих моделей.

[8] https://github.com/Intelligent-Systems-Phystech/Pilkevich-BS-Thesis

Список литературы

[1] Tom Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared D Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, Amanda Askell, Sandhini Agarwal, Ariel Herbert-Voss, Gretchen Krueger, Tom Henighan, Rewon Child, Aditya Ramesh, Daniel Ziegler, Jeffrey Wu, Clemens Winter, Chris Hesse, Mark Chen, Eric Sigler, Mateusz Litwin, Scott Gray, Benjamin Chess, Jack Clark, Christopher Berner, Sam McCandlish, Alec Radford, Ilya Sutskever, and Dario Amodei. Language models are few-shot learners. 33:1877–1901, 2020.

[2] Colin Raffel, Noam Shazeer, Adam Roberts, Katherine Lee, Sharan Narang, Michael Matena, Yanqi Zhou, Wei Li, and Peter J. Liu. Exploring the limits of transfer learning with a unified text-to-text transformer. *Journal of Machine Learning Research*, 21(140):1–67, 2020.

[3] Alexis Conneau, Kartikay Khandelwal, Naman Goyal, Vishrav Chaudhary, Guillaume Wenzek, Francisco Guzmán, Edouard Grave, Myle Ott, Luke Zettlemoyer, and Veselin Stoyanov. Unsupervised cross-lingual representation learning at scale. *arXiv preprint arXiv:1911.02116*, 2019.

[4] Eleftheria Briakou, Sweta Agrawal, Joel Tetreault, and Marine Carpuat. Evaluating the evaluation metrics for style transfer: A case study in multilingual formality transfer. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, pages 1321–1336, Online and Punta Cana, Dominican Republic, November 2021. Association for Computational Linguistics.

[5] Danilo Croce, Giuseppe Castellucci, and Roberto Basili. Gan-bert: Generative adversarial learning for robust text classification with a bunch of labeled examples. pages 2114–2119, jul 2020.

[6] David Donahue and Anna Rumshisky. Adversarial text generation without reinforcement learning. *CoRR*, abs/1810.06640, 2018.

[7] Lantao Yu, Weinan Zhang, Jun Wang, and Yong Yu. Seqgan: Sequence generative adversarial nets with policy gradient. *Proceedings of the AAAI Conference on Artificial Intelligence*, 31(1), Feb. 2017.

[8] Matt J Kusner and José Miguel Hernández-Lobato. Gans for sequences of discrete elements with the gumbel-softmax distribution. *arXiv preprint arXiv:1611.04051*, 2016.

[9] Maja Popović. chrf: character n-gram f-score for automatic mt evaluation. pages 392–395, sep 2015.

[10] Rico Sennrich, Barry Haddow, and Alexandra Birch. Neural machine translation of rare words with subword units. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1715–1725, Berlin, Germany, August 2016. Association for Computational Linguistics.

[11] Chi Sun, Xipeng Qiu, Yige Xu, and Xuanjing Huang. How to fine-tune bert for text classification? In *China national conference on Chinese computational linguistics*, pages 194–206. Springer, 2019.

[12] Alexey Tikhonov and Ivan P. Yamshchikov. What is wrong with style transfer for texts? *ArXiv*, abs/1808.04365, 2018.

[13] Zhilin Yang, Zihang Dai, Yiming Yang, Jaime Carbonell, Russ R Salakhutdinov, and Quoc V Le. Xlnet: Generalized autoregressive pretraining for language understanding. In H. Wallach, H. Larochelle, A. Beygelzimer, F. d'Alché-Buc, E. Fox, and R. Garnett, editors, *Advances in Neural Information Processing Systems*, volume 32. Curran Associates, Inc., 2019.

[14] Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. BERT: Pre-training of deep bidirectional transformers for language understanding. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 4171–4186, Minneapolis, Minnesota, June 2019. Association for Computational Linguistics.

[15] Varvara Logacheva, Daryna Dementieva, Irina Krotova, Alena Fenogenova, Irina Nikishina, Tatiana Shavrina, and Alexander Panchenko. A study on manual and automatic evaluation for text style transfer: The case of detoxification. In *Proceedings of the 2nd Workshop on Human Evaluation of NLP Systems (HumEval)*, pages 90–101, Dublin, Ireland, May 2022. Association for Computational Linguistics.

[16] Ian J. Goodfellow, Jean Pouget-Abadie, Mehdi Mirza, Bing Xu, David Warde-Farley, Sherjil Ozair, Aaron Courville, and Yoshua Bengio. Generative adversarial nets. In *Proceedings of the 27th International Conference on Neural Information Processing Systems - Volume 2*, NIPS'14, page 2672–2680, Cambridge, MA, USA, 2014. MIT Press.

[17] `https://russe.nlpub.org/2022/tox/`. Russian Text Detoxification Based on Parallel Corpora.