Федеральное государственное автономное образовательное
учреждение высшего образования
«Национальный исследовательский университет
«Высшая школа экономики»

Факультет компьютерных наук
Основная образовательная программа
Прикладная математика и информатика

ВЫПУСКНАЯ КВАЛИФИКАЦИОННАЯ РАБОТА

Исследовательский проект на тему
"Атаки на мультиязычные модели"

Выполнил студент группы 171, 4 курса,
Биршерт Алексей Дмитриевич

Руководитель ВКР:
Доцент,
Департамент больших данных и информационного поиска
Артемова Екатерина Леонидовна

Москва 2021

Содержание

1 Введение **4**

2 Обзор литературы **5**

2.1 Мультиязычные модели 5

2.2 Классификация интентов и заполнение слотов 5

2.3 Адверсариальные атаки на мультиязычные модели 5

2.4 Машинный перевод и выравнивание слов 5

3 Основная часть **6**

3.1 Обучение моделей на датасете MultiAtis++ 6

3.1.1 Датасет 6

3.1.2 Архитектура модели 7

3.1.3 Обучение 7

3.2 Адверсариальные атаки 8

3.2.1 Общий вид атаки 9

3.2.2 Word level атака 9

3.2.3 Phrase-level атака 10

3.3 Метод адверсариального предобучения для защиты от адверсариальных атак 10

3.3.1 Генерация адверсариальной выборки 11

3.3.2 Дообучение тела модели 12

3.3.3 Генерация адверсариальной выборки 13

3.4 Результаты 13

3.4.1 Кросс-язычные знания в моделях 13

3.4.2 Качество моделей после адверсариальных атак 14

3.4.3 Влияние метода адверсариального предобучения 15

4 Заключение **16**

Список литературы **17**

Приложения **18**

Приложение А. Алгоритм замены слотов в атаке 18

Аннотация

Какие-то слова в абстракте. Какие-то слова в абстракте.
Ссылка на гитхаб с проектом - https://github.com/birshert/attack-lang-models.

Ключевые слова—Ключевые слова

Some words in abstract. Some words in abstract.
Github project link - https://github.com/birshert/attack-lang-models.

Keywords—Keywords

1 Введение

Последние несколько лет стали прорывными в области мультиязычных моделей и их обобщающей способности для других языков [6, 10]. Огромные мультиязычные модели выучивают универсальные языковые представления, что помогает им демонстрировать удивительные способности к переносу знаний с одного языка на другой. Простое дообучение предобученных моделей для какой-либо задачи на языке с большим количеством данных позволяет достичь хорошего качества на других языках.

Однако простой перенос между языками недостаточен для систем обработки естественного языка для понимания мультиязычных пользователей. Во многих сообществах в мире достаточно часто явление смешения кодов. Смешение кодов — это процесс, когда человек спонтанно смешивает различные языки внутри одного предложения или фразы. Такой феномен может проявляться как в письменной, так и в устной речи. Таким образом, важно сделать языковую модель устойчивой к смешению языков, чтобы модель адекватно работала со входными данными.

Несмотря на то, что реальные данные со смешением языков очень важны для оценки способности языковых моделей работы со смешением кодов, такие данные очень тяжело собирать и размечать в большом количестве.

В своей работе мы предполагаем, что качество моделей на адверсариальных атаках может служить нижней оценкой на реальное качество модели. Если языковая модель успешно справляется с адверсариальными пертурбациями со смешением кодов, то и в реальной жизни она будет успешно обрабатывать данные от мультиязычных пользователей. Таким образом, мы в своей работе:

- Предлагаем две адверсариальные атаки по методу серого ящика — во время атаки мы имеем доступ к ошибке модели на заданных данных. Мы проводим атаки на мультиязычные модели для задачи одновременного детектирования намерений пользователя и заполнения слотов для

диалоговых помощников, направленных на выполнение конкретной задачи. Насколько нам известно, это одни из первых мультиязычных адверсариальных атак на данную задачу.

- Предлагаем метод адверсариального предобучения и показываем, что он увеличивает качество моделей на наших атаках.
- Дополнительно исследуем перенос знаний для задачи одновременной классификации интентов и разметки слотов в предложении.

Все свои эксперименты мы будем проводить с современными мультиязычными моделями - m-BERT [2] и XLM-RoBERTa [1]. В качестве датасета мы будем использовать корпус MultiAtis++ [11].

2 Обзор литературы

2.1 Мультиязычные модели

BERT - [2, 6, 10] , XLM-RoBERTa - [1],

2.2 Классификация интентов и заполнение слотов

[9]

2.3 Адверсариальные атаки на мультиязычные модели

[8, 5]

2.4 Машинный перевод и выравнивание слов

Перевод - [4].

Выравнивание - [3].

3 Основная часть

3.1 Обучение моделей на датасете MultiAtis++

В своей работе мы обучаем языковые модели решать задачу задачи одновременного детектирования намерений пользователя и заполнения слотов для диалоговых помощников, направленных на выполнение конкретной задачи. Эта задача заключается в классификации предложений и всех слов в предложении.

3.1.1 Датасет

В качестве датасета в своей работе мы выбрали датасет MultiAtis++ [11]. В этом датасете представлены семь языков из трёх языковых семей — Индо-Европейская (английский, немецкий, французский, испанский, португальский), Японо-рюкюская (японский) и Сино-тибетская (китайский). Датасет является параллельным корпусом для задачи классификации интентов и разметки слотов - в 2020 году он был переведён с английского языка на остальные шесть. В обучающей выборке содержится 4978 предложений для каждого языка, в тестовой 893 предложения для каждого языка.

Каждый объект в датасете состоит из предложения, меток слов в BIO формате и интента. Перед началом работы с датасетом мы произвели предварительную очистку — убрали из обучающей и тестовой выборок объекты, для которых на любом из семи языков количество слов и количество слотов не совпадали. Таким образом, в обучающей выборке осталось 4884 объекта для каждого языка, в тестовой выборке 755 объектов для каждого языка. Для составления списка используемых слотов и интентов использовалась обучающая выборка на английском языке. Мы использовали 121 различную метку слотов и 23 различных метки интентов. Список id используемых объектов, а также списки используемых слотов и интентов можно найти в приложении.

3.1.2 Архитектура модели

В своей работе мы решаем задачу одновременной классификации интентов и разметки слотов в предложении с помощью одной модели. Модель имеет два выхода, первый предсказывает интенты, второй предсказывает метки слов. В качестве рассматриваемых архитектур были выбраны модели m-BERT [2] и XLM-RoBERTa [1]. Обе эти модели являются одними из самых сильных мультиязычных моделей на текущий момент. Каждая из них предобучена на более чем ста языках.

Обозначим количество блоков Трансформера за L, размер скрытых представлений за H и количество голов с внутренним вниманием за A. Тогда в используемой нами модели m-BERT L = 12, H = 768, A = 12, а суммарное количество параметров 110 миллионов. В используемой нами модели XLM-RoBERTa L = 12, H = 768, A = 12, а суммарное количество параметров 270 миллионов.

3.1.3 Обучение

В своей работе мы будем сравнивать модели, обученные на всей обучающей выборке и только на части обучающей выборки на английском языке. Таким образом мы сможем проверить гипотезу о наличии кросс-язычных знаний у моделей. Тестовая выборка, которая будет нас интересовать в данном контексте состоит из всех семи языков, но мы оцениваем качество на каждом языке отдельно.

Каждая из моделей обучалась с одинаковыми гиперпараметрами - 10 эпох на обучающей выборке с длиной шага обучения 10^{-5} и размером батча в 64 объекта. В качестве функции ошибки использовалась кросс-энтропия:

$$L = -\frac{1}{n}\sum_{i=1}^{n}\left[y \log\left(\hat{y}\right)\right] \tag{1}$$

В своей работе мы будем использовать следующие метрики качества:

- Доля предложений, в которых правильно классифицирован интент:

$$\textbf{Intent accuracy} = \#\text{sentences}\left[(I_{pred} = I_{true})\right] \tag{2}$$

- F1 мера для меток слотов (используется микро-усреднение по всем классам):

$$\textbf{Slots F1 score} = 2 \cdot \frac{\text{precision} \cdot \text{recall}}{\text{precision} + \text{recall}} \tag{3}$$

- Доля предложений, в которых правильно классифицирован интент и верно классифицированы все слоты:

$$\textbf{Semantic accuracy} = \#\text{sentences}\left[(I_{pred} = I_{true}) \wedge (S_{pred} = S_{true})\right] \tag{4}$$

3.2 Адверсариальные атаки

В своей работе мы предлагаем два варианта gray-box адверсариальных атак — во время выполнения атаки мы имеем доступ к ошибке модели. Мы стремимся создать атаку такого рода, чтобы результирующая адверсариальная пертурбация предложения была как можно ближе к реалистичным предложениям со смешением кодов. Для этого мы заменяем часть токенов в предложении на их эквиваленты из других языков. Оценка качества на таких адверсариальных атаках может выступать в роли оценки снизу на качество соответствующих моделей в аналогичных задачах при наличии реального смешения кодов во входных данных.

Так как большинство людей, которые могут использовать смешение кодов в своей речи билингвы, то в основном смешение кодов происходит между парой языков [7]. Таким образом, в своей работе мы предлагаем анализировать атаки состоящие во встраивании одного языка в другой.

3.2.1 Общий вид атаки

Общий принцип атаки одинаковый для обоих предлагаемых вариантов. Разница между методами заключается в способе генерации кандидатов на замену токену на $i-$ой позиции. В своей работе мы предлагаем следующий вид атаки — пусть мы имеем целевую модель, пару пример-метка и встраиваемый язык (1). Тогда мы перебираем токены в предложении в случайном порядке и стремимся заменить токен на его эквивалент из встраиваемого языка. Если это приведёт к увеличению ошибки модели, то мы заменяем токен на предложенного кандидата.

Algorithm 1 Общая схема адверсариальной атаки

Require: Пара пример-метка x, y; целевая модель $\mathcal{M}$; встраиваемый язык $\mathbb{L}$
Ensure: Адверсариальный пример x'

```
L_x = GetLoss(M, x, y)
for i in permutation(len(x)) do
    Candidates = GetCandidates(M, x, y, token_id = i)
    Losses = GetLoss(M, Candidates)
    if Candidates and max(Losses) > L_x then
        L_x = max(Losses)
        x, y = Candidates[argmax(Losses)]
    end if
end for
    return x
```

3.2.2 Word level атака

Первый предлагаемый нами вариант атаки заключается в генерации эквивалентов из других языков с помощью перевода токенов на соответствующие языки. Этот вариант является грубой оценкой снизу, так как он не учитывает контекста предложений и не учитывает многозначность слов.

Для перевода слов на другие языки мы используем модель машинного перевода M2M 100 от компании Facebook [4]. Она содержит 418 миллионов параметров.

Algorithm 2 Word-level атака

Require: Словарь переводов с исходного на встраиваемый язык $\mathbb{T}$

```
function GetCandidates(M, x, y, token_id)
    if x[token_id] in T[L] then
        tokens = T[L][x[token_id]]
        x[token_id] = tokens
        y[token_id] = ExtendSlotLabels(y[token_id], len(tokens))
    end if
    return x, y
end function
```

3.2.3 Phrase-level атака

Второй предлагаемый нами вариант атаки заключается в генерации эквивалентов из других языков с помощью построения выравниваний между предложениями на разных языков. Кандидаты для каждого токена определяются как токены из предложения на встраиваемом языке, в которые был выровнен токен.

Для построения выравниваний мы используем модель awesome-align на основе m-BERT [3].

Algorithm 3 Word-level атака

Require: Выравнивание предложения на исходном языке к предложению на целевом языке $\mathbb{A}$

```
function GetCandidates(M, x, y, token_id)
    if x[token_id] in A[L] then
        tokens = A[L][x[token_id]]
        x[token_id] = tokens
        y[token_id] = ExtendSlotLabels(y[token_id], len(tokens))
    end if
    return x, y
end function
```

3.3 Метод адверсариального предобучения для защиты от адверсариальных атак

В своей работе мы предлагаем метод защиты от предложенных выше адверсариальных атак. Гипотеза заключается в том, что данный метод позво-

лит увеличить качество не только на адверсариальных пертурбациях, но и на реальных данных со смешением кодов.

Предлагаемый нами метод адверсариального предобучения состоит из нескольких шагов:

1 Генерация выборки для задачи маскированного моделирования языка.

2 Дообучение тела мультиязычной модели на сгенерированной выборке в режиме предсказания маскированных токенов.

3 Загрузка дообученного тела модели перед началом обучения для задачи одновременного заполнения слотов и классификации интентов.

Algorithm 4 Генерация адверсариальной выборки

Require: Обучающая выборка датасета X, набор встраиваемых языков $\mathbb{L}_1, \ldots \mathbb{L}_n$
Ensure: Адверсариальная выборка X'

```
X' = [ ]
for 𝕃 in 𝕃_1, ... 𝕃_n do
    for x in X do
        for i in permutation(len(x)) do
            Candidates = GetCandidates(ℳ, x, y, token_id = i)
            if Candidates and 𝒰(0, 1) > 0.5 then
                x, _ = random.choice(Candidates)
            end if
        end for
        X'.append(x)
    end for
end for
return X'
```

3.3.1 Генерация адверсариальной выборки

Для генерации выборки используется адаптация алгоритма phrase-level адверсариальной атаки. Разница заключается в том, что токены заменяются на их эквиваленты с некоторой вероятностью. Таким образом, для генерации выборки не требуется обученная модель.

Выборка является конкатенацией сгенерированных выборок для всех шести языков кроме английского представленных в датасете. Каждая из подвыборок генерируется встраиванием целевого языка в обучающую выборку датасета MultiAtis++ на английском языке (4). Псевдокод функции GetCandidates представлен в секции про атаки (3).

После генерации у нас получается 6 подвыборок по 4884 предложения в каждой. Итоговая выборка состоит из 29304 предложений, мы делим эту выборку в отношении 9 к 1 на обучающую и тестовую.

3.3.2 Дообучение тела модели

После генерации адверсариальной выборки мы дообучаем предобученную мультиязычную модель на этой выборке. Модель обучается в режиме задачи маскированного моделирования языка.

Для обучения модели для такой задачи мы маскируем 15% токенов и предсказываем их с помощью модели. 80% маскированных токенов заменяются на токен маски, 10% заменяются на случайные слова из словаря, остальные 10% остаются неизменными. Мы дообучаем обе мультиязычные модели m-BERT и XLM-RoBERTa с одинаковыми гиперпараметрами - 10 эпох с размером батча 64 и длиной шага 10^{-5}. После дообучения мы сохраняем тело модели для дальнейшего использования.

3.3.3 Генерация адверсариальной выборки

3.4 Результаты

3.4.1 Кросс-язычные знания в моделях

	xlm-r	xlm-r en	xlm-r adv	xlm-r en + adv
Intent accuracy	0.980	0.902	0.981	0.928
Slot F1 score	0.944	0.870	0.947	0.888
Semantic accuracy	0.826	0.559	0.833	0.613
Loss	0.317	0.729	0.320	0.621

Таблица 1: Сравнение моделей XLM-R между собой на тестовой выборке (английский язык)

	m-bert	m-bert en	m-bert adv	m-bert en + adv
Intent accuracy	0.979	0.952	0.975	0.959
Slot F1 score	0.947	0.899	0.950	0.900
Semantic accuracy	0.854	0.672	0.861	0.674
Loss	0.353	0.584	0.326	0.567

Таблица 2: Сравнение моделей M-BERT между собой на тестовой выборке (английский язык)

	xlm-r	xlm-r en	xlm-r adv	xlm-r en + adv
Intent accuracy	0.969 ± 0.004	0.840 ± 0.044	0.970 ± 0.004	0.860 ± 0.043
Slot F1 score	0.928 ± 0.011	0.669 ± 0.063	0.930 ± 0.013	0.675 ± 0.113
Semantic accuracy	0.775 ± 0.044	0.181 ± 0.107	0.781 ± 0.048	0.245 ± 0.167
Loss	0.399 ± 0.055	1.498 ± 0.368	0.409 ± 0.063	1.453 ± 0.525

Таблица 3: Сравнение моделей XLM-R между собой на тестовой выборке (все языки кроме английского)

	m-bert	m-bert en	m-bert adv	m-bert en + adv
Intent accuracy	0.964 ± 0.008	0.828 ± 0.043	0.967 ± 0.006	0.837 ± 0.072
Slot F1 score	0.927 ± 0.020	0.616 ± 0.093	0.929 ± 0.015	0.576 ± 0.101
Semantic accuracy	0.776 ± 0.064	0.204 ± 0.103	0.779 ± 0.055	0.219 ± 0.117
Loss	0.425 ± 0.093	1.584 ± 0.348	0.382 ± 0.057	1.794 ± 0.768

Таблица 4: Сравнение моделей M-BERT между собой на тестовой выборке (все языки кроме английского)

3.4.2 Качество моделей после адверсариальных атак

Word-level атака значительно уменьшила качество для обеих моделей, причем M-BERT справился с этой атакой хуже.

	xlm-r	xlm-r en	xlm-r adv	xlm-r en + adv
Intent accuracy	0.876 ± 0.034	0.721 ± 0.086	0.888 ± 0.033	0.769 ± 0.079
Slot F1 score	0.642 ± 0.083	0.550 ± 0.068	0.640 ± 0.088	0.554 ± 0.076
Semantic accuracy	0.177 ± 0.101	0.065 ± 0.063	0.174 ± 0.102	0.101 ± 0.070
Loss	2.662 ± 0.737	3.234 ± 0.807	2.553 ± 0.669	2.905 ± 0.562

Таблица 5: Сравнение моделей XLM-R после word-level атаки

	m-bert	m-bert en	m-bert adv	m-bert en + adv
Intent accuracy	0.863 ± 0.025	0.771 ± 0.028	0.893 ± 0.017	0.819 ± 0.047
Slot F1 score	0.553 ± 0.098	0.447 ± 0.084	0.581 ± 0.085	0.455 ± 0.069
Semantic accuracy	0.117 ± 0.075	0.055 ± 0.051	0.155 ± 0.085	0.081 ± 0.054
Loss	3.169 ± 0.689	3.338 ± 0.667	2.860 ± 0.687	2.959 ± 0.574

Таблица 6: Сравнение моделей M-BERT после word-level атаки

	xlm-r	xlm-r en	xlm-r adv	xlm-r en + adv
Intent accuracy	0.949 ± 0.011	0.727 ± 0.131	0.952 ± 0.011	0.827 ± 0.035
Slot F1 score	0.708 ± 0.139	0.584 ± 0.111	0.716 ± 0.147	0.621 ± 0.146
Semantic accuracy	0.368 ± 0.161	0.106 ± 0.071	0.392 ± 0.156	0.214 ± 0.133
Loss	2.032 ± 1.156	2.860 ± 0.839	2.032 ± 1.233	2.113 ± 0.637

Таблица 7: Сравнение моделей XLM-R после phrase-level атаки

	m-bert	m-bert en	m-bert adv	m-bert en + adv
Intent accuracy	0.941 ± 0.006	0.829 ± 0.018	0.951 ± 0.005	0.847 ± 0.054
Slot F1 score	0.700 ± 0.127	0.538 ± 0.097	0.725 ± 0.142	0.578 ± 0.132
Semantic accuracy	0.345 ± 0.128	0.110 ± 0.055	0.424 ± 0.159	0.214 ± 0.116
Loss	2.131 ± 1.138	2.463 ± 0.585	1.970 ± 1.196	2.159 ± 0.755

Таблица 8: Сравнение моделей M-BERT после phrase-level атаки

Как видно по таблицам (5), (6), (7), (8)

3.4.3 Влияние метода адверсариального предобучения

4 Заключение

AAAAAAAAAAAAAAAAAA FUCK ME

Список литературы

[1] Alexis Conneau и др. «Unsupervised Cross-lingual Representation Learning at Scale». В: *ACL*. 2020.

[2] Jacob Devlin и др. «BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding». В: *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)*. 2019, с. 4171—4186.

[3] Zi-Yi Dou и Graham Neubig. «Word Alignment by Fine-tuning Embeddings on Parallel Corpora». В: *EACL*. 2021.

[4] Angela Fan и др. «Beyond English-Centric Multilingual Machine Translation». В: *ArXiv* abs/2010.11125 (2020).

[5] Jitin Krishnan и др. «Multilingual Code-Switching for Zero-Shot Cross-Lingual Intent Prediction and Slot Filling». В: *ArXiv* abs/2103.07792 (2021).

[6] Chi-Liang Liu и др. «What makes multilingual BERT multilingual?» В: *ArXiv* abs/2010.10938 (2020).

[7] Shana Poplack, DAVID SANKOFF и CHRISTOPHER MILLER. «The social correlates and linguistic processes of lexical borrowing and assimilation». В: *Linguistics* 26 (1988), с. 47—104.

[8] Samson Tan и Shafiq Joty. «Code-Mixing on Sesame Street: Dawn of the Adversarial Polyglots». В: *ArXiv* abs/2103.09593 (2021).

[9] H. Weld и др. «A survey of joint intent detection and slot-filling models in natural language understanding». В: *ArXiv* abs/2101.08091 (2021).

[10] Shijie Wu и Mark Dredze. «Beto, Bentz, Becas: The Surprising Cross-Lingual Effectiveness of BERT». В: *EMNLP/IJCNLP*. 2019.

[11] Weijia Xu, Batool Haider и Saab Mansour. «End-to-End Slot Alignment and Recognition for Cross-Lingual NLU». В: *ArXiv* abs/2004.14353 (2020).

Приложения

Приложение А. Алгоритм замены слотов в атаке

Algorithm 5 Алгоритм замены слотов в атаке

```
function ExtendSlotLabels(slot_label, num_tokens)
    slot_labels = [slot_label]
    if num_tokens > 1 then
        if slot_label.startswith('B') then
            slot_labels += ['I' + slot_label[1:]] · (num_tokens - 1)
        else
            slot_labels ·= num_tokens
        end if
    end if
    return slot_labels
end function
```