Оглавление

Введение

Данная работа посвящена интерактивному подходу к решению задачи распознавания диктора. Оригинальный метод был предложен в [1], и значительная часть работы посвящена его описанию и практической реализации. С момента публикации этой статьи (2020 год) уже прошло достаточное количество времени, но она не стала популярной — по данным *Google Scholar* на момент написания этого отчёта она была процитирована 6 раз[1]. Тем не менее, нам (автору дипломной работы, его научному руководителю и коллегам из лаборатории Huawei CBG AI) она показалась заслуживающей внимания. На это есть ряд причин.

В первую очередь стоит отметить оригинальность предложенного подхода. Исторически большинство работ, в той или иной степени затрагивающие задачу распознавания диктора, посвящены способам как можно лучше определять диктора на основе уже существующих аудиозаписей. Рассматриваемая работа ставит проблему иначе — какие слова или фразы должен произнести диктор, чтобы уже существующая система смогла распознать его как можно быстрее и надёжнее. Чем-то такой подход напоминает концепцию активного обучения (*англ.* active learning) — разметки только тех данных, которые являются наиболее важными для решающей функции.

Другой причиной интереса к работе стала возможность её потенциального использования в конечном продукте — системе аутентификации пользователя на мобильном устройстве или персональном ассистенте. Предполагается, что такая система будет спрашивать пользователя произнести ту или иную фразу, пока она не станет уверена, что перед ней действительно находится настоящий владелец прибора. В таком случае логично делать не случайные запросы, а такие, которые позволят системе как можно быстрее идентифицировать пользователя.

Более подробное описание метода дано в главе 1. Следующая глава посвящена практической реализации описанного метода и полученным результатам. Глава 3 в свою очередь посвящена модификациям оригинального подхода, направленными на повышение точности и адаптации метода под

[1]Из этих цитат 1 приходится на кандидатскую диссертацию её первого автора.

сформулированную выше практическую задачу.

1. Распознавание диктора в интерактивном режиме

1.1. Задача распознавания диктора

Распознавание диктора является одной из задач обработки речи — обширного научного и исследовательского направления с долгой и богатой историей. Как и в случае со многими другими научными направлениеями, в последние годы обработка речи стала активно использовать методы машинного обучения, в частности нейросетевые модели.

Задача распознавания диктора, как нетрудно понять из названия, заключается в определении личности человека по аудиозаписи его речи. Если говорить чуть более строго, задачей является сопоставление некоторой аудиозаписи речи неизвестного человека с некоторым набором дикторов. В случае решения задачи *идентификации* этот набор состоит из нескольких дикторов, при этом мы точно знаем, что один из них произнес анализируемую нами речь. Соответственно, в таком случае задачей системы является правильный выбор диктора. В случае решения задачи *верификации* нам известна информация только об одном диктора. Таким образом, от нас требуется определить, произнёс ли он речь на предоставленной нам аудиозаписи.

1.2. Интерактивный режим

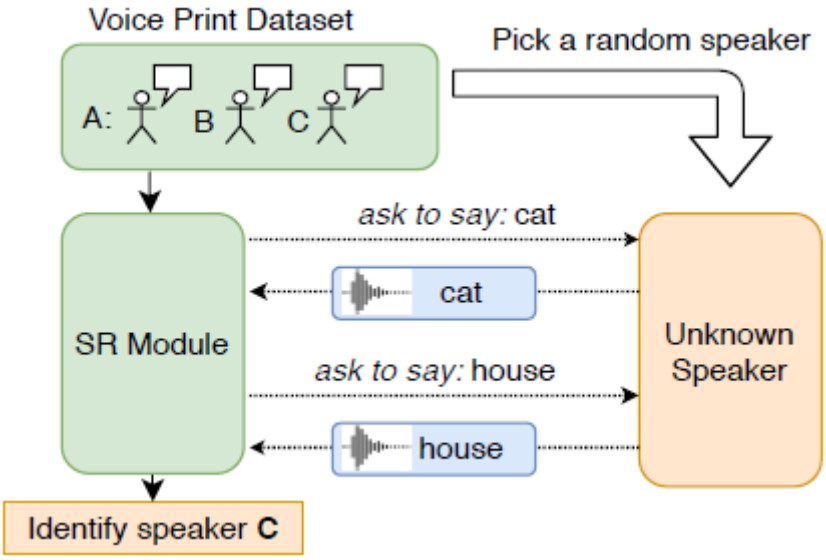

Рис. 1. Схема интерактивной игры по определению диктора [1]

2. Детали реализации и результаты

2.1. Данные для обучения и извлечение эмбеддингов

Здесь мы практически полностью повторяем описанный в [1] подход. Единственным (но очень существенным) отличием является использованная размерность эмбеддингов. Перед тем как перейти к обсуждению этого момента, расскажем про исходные данные.

Итак, для обучения и тестирования моделей мы использовали датасет TIMIT[2]. Он составлен из аудиозаписей речи 630 дикторов, говорящих на 8 основных диалектах американского английского языка. Эти дикторы поделены на обучающую (`train`) и тестовую (`test`) выборки, в первую входят 468 дикторов, во вторую — 162. Для обучения нейросетевых моделей мы также создавали валидационную выборку, в которую выделялись 20% дикторов из обучающей.

Каждый из дикторов произносит 10 фонетически насыщенных предложений. При этом 2 из 10 предложений являются общими для всех дикторов [2] , остальные 8 уникальны для каждого диктора. Такое разделение позволя-

[2]Общие предложения:

ет без особых затруднений подготовить данные, необходимые для описанной в 1.2 игры:

- 2 общих предложения можно использовать для получения аудиозаписей слов. Для этого разделим аудиозаписи этих предложений по временным отметкам, предоставленным создателями датасета. В результате получим 20 аудиозаписей слов[3] для каждого диктора.

- 8 уникальных для каждого диктора предложений можно использовать для получения голосовых подписей — эмбеддингов дикторов — просто при помощи усреднения эмбеддингов аудиозаписей этих предложений.

В качестве векторов признаков использовались эмбеддинги `x-vector` [3]. Весь процесс преобразования аудиозаписей в векторы признаков осуществлялся с помощью библиотеки Kaldi [4]. На первом этапе рассчитывались мел-частотные кепстральные коэффициенты[4] и производилось детектирование голосовой активности (*англ.* VAD — voice activity detection). Полученные векторы признаков поступали на вход предобученной нейронной сети [5]. В качестве эмбеддингов использовались данные со второго 512-мерного слоя.

Здесь, как уже было сказано ранее, мы отступаем от оригинальной работы [1], где использовались 128-мерные эмбеддинги. На это есть две причины. Во-первых, из приведенных в [1] комментариев неочевидно[5], как производилось понижение размерности. Во-вторых, мотивация такого преобразования тоже неочевидна. Уже первые проведенные нами эксперименты показали, что при использовании 512-мерных эмбеддингов точность идентификации оказывается существенно выше приведенных в [1] значений.

She had your dark suit in greasy wash water all year.
Don't ask me to carry an oily rag like that.

[3]Аналогично [1] мы не используем слово *an*.

[4]Параметры аналогичны использованным в [1] и определяются требованиями предобученной модели.

[5]Цитата: *We then process the MFCCs features through a pretrained X-Vector network to obtain a high quality voice embedding of fixed dimension 128, where the X-Vector network is trained on augmented Switchboard, Mixer 6 and NIST SREs.*

2.2. Обучение `Guesser`

Первой обучается нейронная сеть `Guesser`, выполняющая выбор из K дикторов при помощи T аудиозаписей произнесенных слов. Как уже было сказано ранее, эта нейросеть тренируется в режиме обучения с учителем, дикторы и произносимые слова выбираются случайно, в качестве функции используется кросс-энтропия. Процесс вычисления значения функции потерь для одной игры можно записать следующим образом:

Listing 1. Рассчёт функции потерь `Guesser`

```
speaker_ids = speakers.sample(size=K)
G = voice_prints.get(speaker_ids)
target = randrange(0, K)
word_inds = randrange(0, V, size=T)
X = word_vocab.get(speaker=speaker_ids[target],
                   words=word_inds)
probabilities = guesser.forward(G, X)
loss = cross_entropy(probabilities, target)
```

Из-за увеличения относительно [1] размерности эмбеддингов пропорционально увеличились и размерности слоёв `Guesser`. Из-за этого нам пришлось изменить гиперпараметры, в частности мы сильно уменьшили темп обучения (*learning rate*).

Как и в оригинальной статье, для сравнения моделей будем строить графики *word* и *guest sweep*. Т. е. будем обучать модель в режиме с $K = 5$ дикторами и $T = 3$ запрашиваемыми словами, а затем будем тестировать её в режимах с отличным числом дикторов или слов. Здесь и далее, если это не оговорено отдельно, для расчёта точности проводятся 20000 игр среди дикторов из тестовой выборки, эксперименты повторяются по 5 раз с различным `seed` генератора случайных чисел.

По приведенным на графиках результатам видно, что увеличение размерности эмбеддингов существенно улучшает точность идентификации, разница особо велика в режимах с большим числом дикторов K.

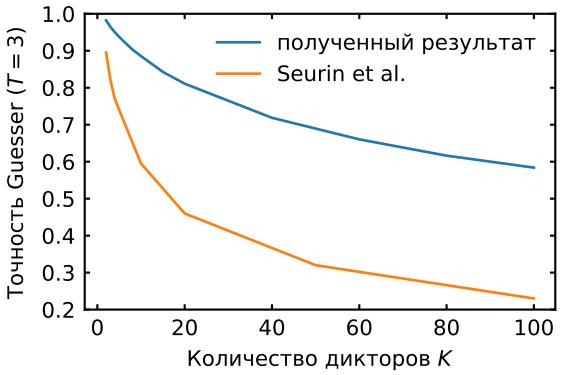

Рис. 2. Зависимость точности `Guesser` обученного нами и авторами [1] от числа дикторов K. Модели обучены в режиме $K = 5$, $T = 3$.

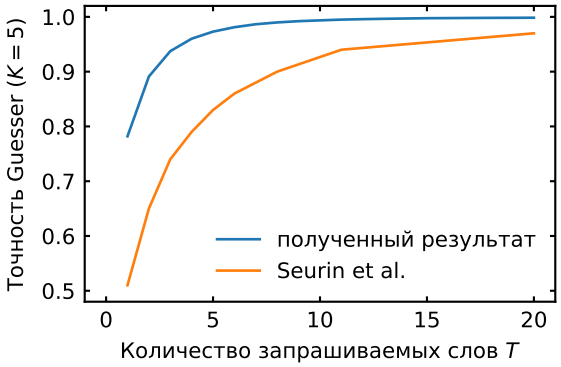

Рис. 3. Зависимость точности `Guesser` обученного нами и авторами [1] от числа запрошенных слов T. Модели обучены в режиме $K = 5$, $T = 3$.

2.3. Обучение `Enquirer`

Для обучения `Enquirer` — модели выбора слов — уже нужна обученная модель `Guesser`. На этом этапе уже используется обучение с подкреплением, псевдокод для 1 игры приведён ниже.

Listing 2. Интерактивная игра для обучения `Enquirer`

```
speaker_ids = speakers.sample(size=K)
G = voice_prints.get(speaker_ids)
target = randrange(0, K)
```

```
g_hat = G.mean(dim=0)
x_i = start_tensor
X = []
for i in range(T):
    probs = enquirer.forward(g_hat, x_i)
    if training:
        word_inds = multinomial(probs).sample()
    else:
        probs[previous_actions] = 0.0
        word_ind = argmax(probs)
    x_i = word_vocab.get(speaker=speaker_ids[target],
                         word=word_ind)
    X.append(x_i)

prediction = guesser.predict(G, X)
reward = 1 if prediction == target else 0
```

Как видно из приведенного псевдокода, награда выдается в том случае, когда `Guesser` правильно угадывает диктора. Для обучения мы использовали алгоритм PPO [6] — здесь мы снова повторяем подход авторов [1]. В целом выбор метода выглядит разумным — PPO зарекомендовал себя как простой и универсальный алгоритм, позволяющий достигать хороших результатов. Однако некоторые особенности нашей задачи — дискретное пространство действий, малая длительность эпизодов — выглядят лучше подходящими для off-policy алгоритмов. К сожалению, у нас не нашлось времени, чтобы проверить эту гипотезу.

Приведенные результаты свидетельствуют о том, что `Enquirer` действительно успешно обучается — точность оказываются заметно выше, чем с случае случайного выбора слов. Как и в случае повышения размерности эмбеддингов, особенно большое различие наблюдается в режимах с большим числом дикторов.

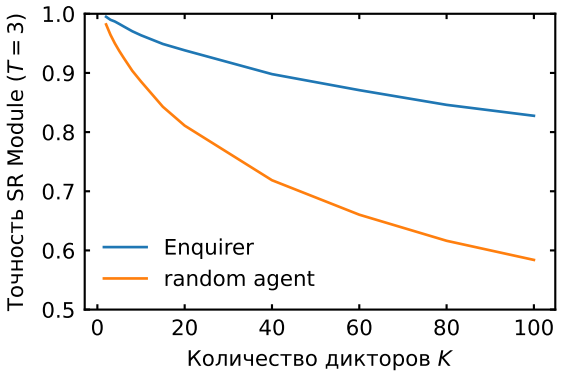

Рис. 4. Зависимость точности SR-систем с различными методами выбора слов — нейросетевым агентов (`Enquirer`) и случайным (random agent) — от числа дикторов K. Модели обучены в режиме $K = 5$, $T = 3$.

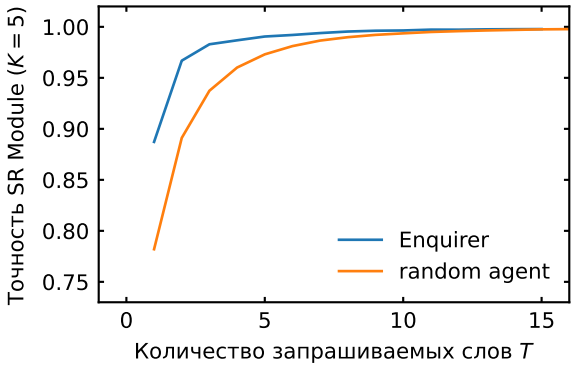

Рис. 5. Зависимость точности SR-систем с различными методами выбора слов — нейросетевым агентов (`Enquirer`) и случайным (random agent) — от числа запрашиваемых слов T. Модели обучены в режиме $K = 5$, $T = 3$.

2.4. Эвристическая модель выбора слов

Очевидно, что агент, выбирающий запрашиваемые слова случайным образом, не является тяжелым противником для нейросетевого агента. Для более трезвой оценки возможностей последнего, логично сравнивать его с каким-то более сложным алгоритмом.

Здесь мы снова немного отходим от оригинальной статьи. И опять основной причиной является тот факт, что в [1] отсутствует точное описание ис-

пользованного в качестве бейзлайна эвристического алгоритма выбора слов. Из приведенных в работе слов[6] общий подход понятен — сэмплирование производится не из всех 20 слов, а из тех, которые в среднем показывают самую высокую точность. При этом остаются непонятными следующие детали:

1. Из скольки слов производится сэмплирование, и меняется ли это число в зависимости от числа запрашиваемых слов T?

2. Производится ли сэмплирование равномерно, или вероятность выбрать слово пропорциональна достигаемой при выборе этого слова средней точности?

Именно такие вопросы возникли у нас при создании эвристического агента. Первым же этапом стала оценка слов — расчёт средней точности, которая достигается случайным агентом в тех играх, когда он выбрал то или иное слово. Для этого мы протестировали `Guesser` в 100000 эпизодов с $K = 5$, и $T = 3$, а также случайным выбором слов без повторений. Мы рассчитывали точность для каждого слова, учитывая только те эпизоды, в которых это слово было выбрано. Фактически мы оценивали условную вероятность связки `Guesser`–случайный агент правильно выбрать диктора при условии, что одно слово уже было выбрано.

После этого мы стали тестировать различные модификации эвристического агента. Как следует из сформулированных выше вопросов, эти агенты отличались числом использованных слов и методом сэмплирования. Эксперименты показали, что наилучшие результаты достигаются при использовании "детерминированного" агента, всегда выбирающего одни и те же слова с наибольшей средней точностью. В таком случае говорить о каком-либо сэмплировании неуместно, поэтому такой агент, по всей видимости, отличается от использованного в оригинальной статье.

В данном случае преимущество `Enquirer` проявляется только в режимах с большим числом дикторов. В стандартном режиме с $K = 5$ дикторами и $T = 3$, в отличие от [1], мы не наблюдаем сколько-нибудь существенной разницы между двумя агентами.

[6]Цитата: *We curated a list of the most discriminant words (words that increase globally the recognition scores) and sample among those instead of the whole list.*

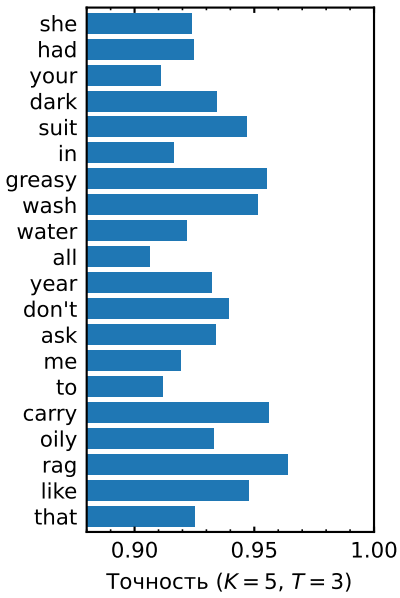

Рис. 6. Средняя точность `Guesser` на валидационной выборке в тех эпизодах, когда соответствующее слово было выбрано (остальные выбирались случайно).

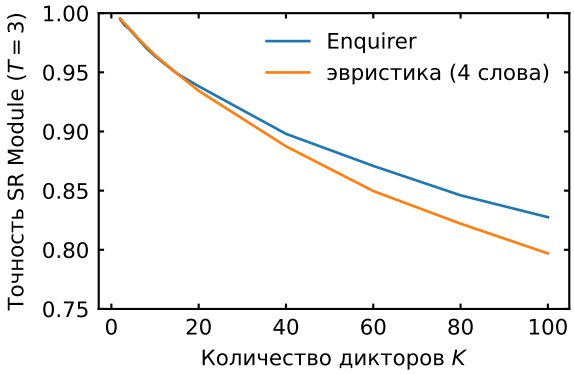

Рис. 7. Зависимость точности SR-систем с различными методами выбора слов — нейросетевым и эвристическим агентами — от числа дикторов K. Модели обучены в режиме $K = 5$, $T = 3$.

В таком случае возникает резонный вопрос — не сходится ли `Enquirer` к такой же политике, что использует эвристический агент? Ответ на этот

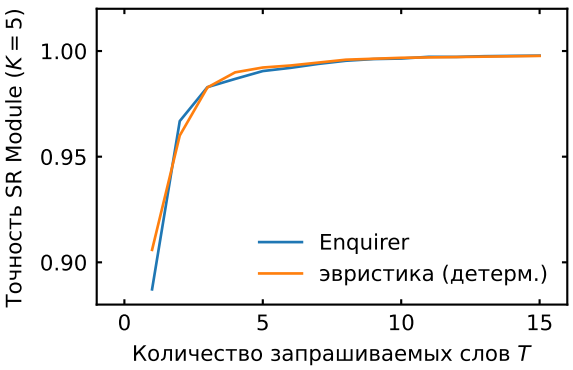

Рис. 8. Зависимость точности SR-систем с различными методами выбора слов — нейросетевым и эвристическим агентами — от числа запрашиваемых слов T. Модели обучены в режиме $K = 5$, $T = 3$.

вопрос — отрицательный, в целом `Enquirer` выбирает из 5 слов (ещё 2 используются редко), в то время как эвристический агент всегда использует 3 тех же слова. Из этого можно предположить, что `Enquirer` обучен недостаточно хорошо, возможно, другие гиперпараметры или алгоритм обучения позволили бы улучшить результаты.

2.5. Обучение в других режимах

Другой логичный вопрос, возникающий при обсуждении графиков *word* и guest sweep — является ли стандартный режим ($K = 5$ дикторов и $T = 3$ запрашиваемых слова) оптимальным для обучения моделей? Не будут ли результаты лучше, если мы будем обучать и тестировать модели в одном и том же режиме? Мы также провели ряд экспериментов и пришли к следующим выводам:

- Общее правило — более тяжелые режимы позволяют улучшить точность. В первую очередь это касается увеличения числа дикторов, ситуация с уменьшением числа слов менее однозначная.
- Основные улучшения наблюдаются в работе `Guesser`, в то же время `Enquirer` оказывается нечувствительным к режиму обучения.

- Обучение при $T = 1$ является специфической задачей. Во-первых, только в этом режиме обученная модель показывает хорошие (лучше чем другие) результаты только в этом же режиме тестирования. Во-вторых, `Enquirer` в среднем показывает плохие результаты в данном режиме, часто уступая максимально простой политике, всегда выбирающей одно и то же слово.

3. Модификации метода

3.1. От идентификации к верификации

В первых двух главах этого отчёта мы изучали предложенную в [1] систему распознавания диктора. С нашей точки зрения у неё есть серьёзный недостаток — она решает задачу *идентификации*, в то время как интересная нам с практической точки зрения система аутентификации пользователя должна решать задачу *верификации*. Ранее мы сформулировали (правда?) тезис о том, что это не является большой проблемой, и переход идентификация–верификация можно выполнить без особых проблем. Обсуждению этого вопроса посвящён данный раздел.

Сначала проговорим, как меняется наша задача. Ранее мы должны были выбрать одного из K дикторов произнесшего T слов, т. е. мы использовали K эмбеддингов дикторов и T эмбеддингов слов. В случае верификации у нас есть только 1 диктор, от нас требуется ответить на вопрос, является ли он человеком, произнесшим услышанную нами речь. Подумаем, какие изменения нам нужно внести в архитектуру использованных нами нейросетей.

В случае `Enquirer` (нейросети для выбора запрашиваемых слов) ответ оказывается предельно простым — нам не нужны никакие изменения. Действительно, сами эмбеддинги диктора на вход этой модели не поступают, используется только их среднее $\hat{g}$, которое в случае верификации будет просто равно эмбеддингу единственного диктора.

Выбор слов	Режим обучения	Точность
случайный		0.895
`Enquirer`	$T = 3$	0.933
эвристика		0.917
случайный		0.913
`Enquirer`	$T = 2$	0.947
эвристика		0.945

Таблица 1. Точность верификации, $T = 3$ запрашиваемых слова

Ситуация с `Guesser` лишь немного сложнее. Т. к. его архитектура позво-

ляет рассматривать игры с произвольным числом дикторов, проблемы возникают только на самом последнем слое, выполняющим операцию `softmax`. На данном этапе у модели (для каждой игры) есть только одно число, которое фактические является некоторой метрикой соответствия между взвешенной суммой эмбеддингов слов $\hat{x}$ и эмбеддингом диктора g. В таком случае для принятия решения о (не-)соответствии речи и диктора логично применить операцию `sigmoid` (логистическую функцию), превращающее эту метрику в число от 0 до 1.

Действительно, такое простое преобразование позволяет получить работающую систему верификации диктора. Полученные результаты приведены в табл. 1. Как и в случае идентификации, обучение в более тяжелом режиме (здесь мы можем только сокращать число запрашиваемых слов) позволяет немного улучшить результаты, но при этом преимущество перед простым эвристическим агентом[7] тоже является минимальным.

3.2. `CodebookEnquirer` — гибкая система выбора слов

Перейдём к обсуждению другой проблемы реализованной нами модели — наличия фиксированного списка слов. Действительно, в качестве одного из преимуществ разрабатываемой системы мы ранее (не) называли возможность делать разнообразные запросы. Однако используемый до данного момента времени вариант `Enquirer` слабо соответствует этому требованию — он осуществляет выбор из 20 слов. Конечно, этот список может быть и больше, для этого просто потребуется больший объём данных для обучения. Но при добавлении любого слова потребуется либо заново обучать `Enquirer`, либо выполнять fine-tuning, что выглядит не самым оптимальным вариантом для готового продукта.

Для решения этой проблемы была разработана архитектура `Codebook Enquirer`. Она представляет собой простую модификацию оригинальной модели:

[7]Здесь он, как и ранее, просто всегда выбирает одни и те же слова, соответствующие наибольшой средней точности на валидационной выборке. Переход к верификации практически никак не меняет градацию слов.

1. “Голова” модели представляет собой `Enquirer`, в котором число выходов равно размерности эмбеддингов, и к ним не применяется операция `softmax`. Таким нехитрым способом мы преобразовали выходы модели из вероятностного распределения по словарю в эмбеддинг запрашиваемого слова.

2. Естественно, стоящая перед `CodebookEnquirer` задача никак не поменялась — у нас все ещё существует некоторый конечный набор слов, из которых на каждом шаге игры нам нужно выбрать одно (или, что лучше, получить распределение). Для этого мы составляем `Codebook` — тензор из эмбеддингов слов, которые рассчитываются как среднее по всем дикторам из обучающей выборки.

3. Наконец, нам нужно как-то сопоставить возвращаемый моделью эмбеддинг с эмбеддингами из `Codebook`. Самый очевидный вариант — просто найти ближайший по L_2-норме. Примерно это мы и делаем, вероятность выбрать i-ое слово из `Codebook` вычисляется по формуле:

$$p_i = \frac{\exp\left(-d_i/T\right)}{\sum_{j=0}^{V} \exp\left(-d_j/T\right)},$$

   где d_i — расстояние[8] между выходным эмбеддингом и i-ым вектором из `Codebook`, T — обучаемый параметр модели, V — размер словаря.

В наших экспериментах такая модификация показала результаты, сравнимые с оригинальной версией `Enquirer` [результаты]. Далее мы решили проверить, возможно ли изменение набора слов без дообучения модели. Для этого мы обучили `CodebookEnquirer` на половине словаря и протестировали его на другой половине. В таком случае мы наблюдали лишь небольшое падение точности, которое, скорее всего, просто объясняется уменьшением размера используемого словаря.

[8] Для численной стабильности мы используем среднеквадратичную ошибку (MSE) вместо L_2-нормы.

3.3. Добавление шума

Следующим экспериментом была проверка того, будет ли работать предложенный подход при наличии фонового шума. Для этого мы выбрали 6 аудиозаписей шума из датасета MUSAN[7] и добавили их случайные фрагменты[9] к аудиозаписям слов. Соотношение сигнал / шум было выбрано равным 3 дБ. При обучении и тестировании моделей тип шума выбирался случайно, но он не менялся в течение игры.

Модель	Идентификация	Верификация
`Guesser`	0.887	0.895
`Guesser` + `Enquirer`	0.946	0.934
`Guesser` + эвристика (3 лучших)	0.957	0.938

Таблица 2. Точность идентификации и верификации в стандартных режимах ($T = 3$ слова, $K = 5$ гостей при идентификации) при добавлении фонового шума.

Полученные результаты приведены в табл. 2. Видно, что добавление шума сделало задачу тяжелее, из-за чего точность SR-систем немного упала. Также любопытно, что простой эвристический агент снова не проиграл `Enquirer`. Причина этого стала понятна после измерения средней точности `Guesser` на аудиозаписях зашумлённых слов: выяснилось, что хотя добавление того или иного типа шума влияет на градацию слов (которую использует эвристический агент), этот эффект невелик. Иными словами, "хорошие" слова, с помощью которых в среднем достигается самая высокая точность распознавания диктора, остались такими же и при добавлении различных типов фонового шума.

3.4. Альтернативные эмбеддинги

Во всех описанных ранее экспериментах для получения эмбеддингов мы использовали `x-vector`[3]. Здесь мы, как и во многих других моментах, повторяем подход авторов оригинальной статьи. Проблема в том, что выбор

[9] Аудиозаписи специально выбирались таким образом, чтобы их случайные короткие фрагменты отличались слабо.

таких старых (2017 год) эмбеддингов выглядел немного странным уже на момент написания оригинальной статьи (2020 год). Сегодня же они выглядят безнадёжно устаревшими.

Поэтому для последнего эксперимента мы проверили, как разработанная SR-модель работает с другими эмбеддингами. Для этого использовалась нейросеть, обученная нашими коллегами из лаборатории Huawei CBG AI на 960 часах аудиозаписей из датасета LibriSpeech[8] и использующая метод контрастного прогнозирующего кодирования[9].

Заключение

все работает, но хотелось бы большего

Список литературы

[1] M. Seurin, F. Strub, P. Preux, and O. Pietquin, "A machine of few words – interactive speaker recognition with reinforcement learning," 2020.

[2] Garofolo, John S., Lamel, Lori F., Fisher, William M., Pallett, David S., Dahlgren, Nancy L., Zue, Victor, and Fiscus, Jonathan G., "TIMIT acoustic-phonetic continuous speech corpus," 1993.

[3] D. Snyder, D. Garcia-Romero, G. Sell, D. Povey, and S. Khudanpur, "X-Vectors: Robust DNN embeddings for speaker recognition," in *2018 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, IEEE, Apr. 2018.

[4] D. Povey, A. Ghoshal, G. Boulianne, L. Burget, O. Glembek, N. Goel, M. Hannemann, P. Motlicek, Y. Qian, P. Schwarz, J. Silovsky, G. Stemmer, and K. Vesely, "The kaldi speech recognition toolkit," in *IEEE 2011 Workshop on Automatic Speech Recognition and Understanding*, IEEE Signal Processing Society, Dec. 2011. IEEE Catalog No.: CFP11SRW-USB.

[5] "SRE16 Xvector Model," 2017.

[6] J. Schulman, F. Wolski, P. Dhariwal, A. Radford, and O. Klimov, "Proximal policy optimization algorithms," 2017.

[7] D. Snyder, G. Chen, and D. Povey, "MUSAN: A Music, Speech, and Noise Corpus," 2015. arXiv:1510.08484v1.

[8] V. Panayotov, G. Chen, D. Povey, and S. Khudanpur, "Librispeech: An asr corpus based on public domain audio books," in *2015 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pp. 5206–5210, 2015.

[9] A. van den Oord, Y. Li, and O. Vinyals, "Representation learning with contrastive predictive coding," 2019.