

Interactive Speaker Recognition

Применение обучения с подкреплением для решения задачи распознавания диктора

Вячеслав Головин
Евгений Шуранов (руководитель)

Huawei CBG AI и ФКН ВШЭ СПб

10.05.2023

Задача: повышение точности систем верификации / идентификации диктора. Такая система может, например, быть использована для подтверждения личности на мобильных устройствах.

Требования к системе:

- короткие запросы (не раздражаем пользователя),
- разнообразные запросы (не боимся спуфинга),
- высокая точность (без комментариев).

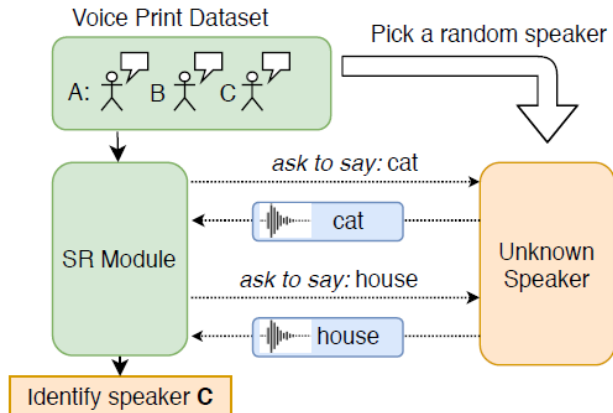
Предлагаемое решение: использование RL-агента для выбора запрашиваемых слов.

Новизна дипломной работы:

- переход от идентификации к верификации,
- более гибкая система для выбора слов.

Interactive Speaker Recognition

Метод был предложен в статье *A Machine of Few Words — Interactive Speaker Recognition with Reinforcement Learning*, Mathieu Seurin et al., INTERSPEECH 2020, arXiv:2008.03127v1.

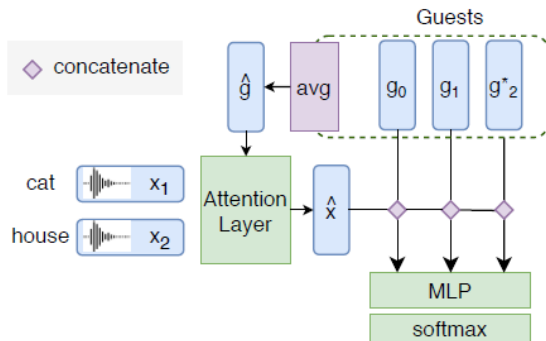


Важные особенности:

- 1 Рассматривается только задача идентификации.
- 2 Набор слов строго фиксирован.
- 3 Разные нейронные сети для двух задач SR Module — запроса слов (Enquirer) и идентификации диктора (Guesser).

Блок Guesser

Архитектура



Входные данные:

- эмбеддинги дикторов
 $G = [g_1; g_2; \dots g_K]$
- эмбеддинги слов
 $X = [x_1; x_2; \dots x_T]$

Выходные данные:

- вероятности
 $\{P(g_i = g^*) \mid i = 1..K\}$

Обозначения

- K количество гостей / дикторов
 T количество запрашиваемых слов

Блок Guesser

Псевдокод 1 итерации обучения

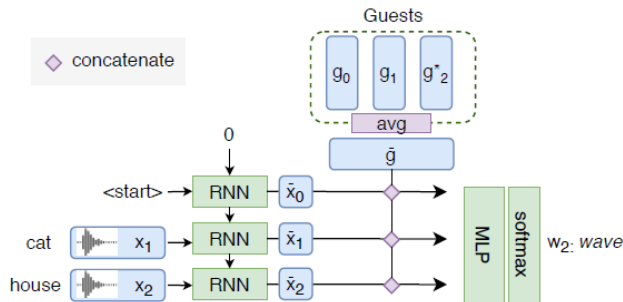
```
speaker_ids = speakers.sample(size=K)
G = voice_prints.get(speaker_ids)
target = randrange(0, K)
word_inds = randrange(0, V, size=T)
X = word_vocab.get(speaker=speaker_ids[target],
                  words=word_inds)
probabilities = guesser.forward(G, X)
loss = cross_entropy(probabilities, target)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

Обозначения

- K количество гостей / дикторов
- T количество запрашиваемых слов
- V размер словаря — число доступных для запроса слов

Блок Enquirer

Архитектура



Входные данные:

- среднее эмб. дикторов

$$\hat{g} = \frac{1}{K} \sum_{i=1}^K g_k$$

- эмбединги слов

$$X = [x_1; x_2; \dots; x_t]$$

Выходные данные:

- вероятность выбрать каждое из слов

Обозначения

- K количество гостей / дикторов
 T количество запрашиваемых слов
 t количество запрошенных слов, $0 \leq t \leq T$

Блок Enquirer

Псевдокод 1 эпизода ISR-игры

```
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:
        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
```

Входные данные

Для обучения использовался датасет **TIMIT**:

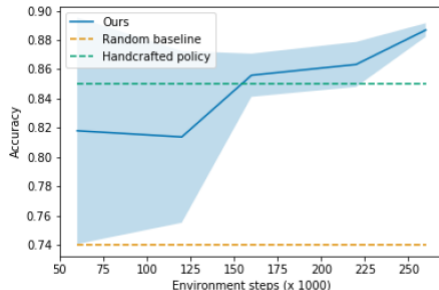
- 630 дикторов из США, 8 акцентов;
- каждый диктор произносит 10 предложений: 8 уникальных и 2 общих.

В качестве эмбедингов использовались **x-vectors**, полученные с помощью нейронной сети, обученной на аугментированных датасетах для распознавания диктора *Switchboard*, *Mixer 6* и *NIST*.

- Эмбединги дикторов g получались с помощью усреднения эмбедингов 8 уникальных предложений.
- Эмбединги слов x извлекались с помощью 2 общих предложений, т.е. сначала вырезались записи одиночных слов, которые затем пропускались через нейронную сеть.

Результаты из статьи

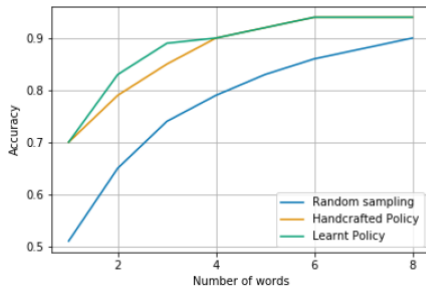
$K = 5$ дикторов и $T = 3$ слова



- RL-агент при выборе запрашиваемых слов учитывает контекст — он опережает не только случайного агента, но и эвристического, выбирающего из подмножества “лучших” слов.

Результаты из статьи

$K = 5$ дикторов и $T = 3$ слова



- RL-агент при выборе запрашиваемых слов учитывает контекст — он опережает не только случайного агента, но и эвристического, выбирающего из подмножества “лучших” слов.
- Преимущество RL-агента невелико и проявляется только при небольшом числе запрашиваемых слов.

Воспроизведение результатов

Основная возникшая проблема — размерность эмбедингов. В статье используются 128-мерные эмбединги, в то время как использованная для их извлечения нейросеть имеет только 512-мерные выходы.

Информации о способе понижения размерности в статье нет. При этом

Извлечение эмбедингов

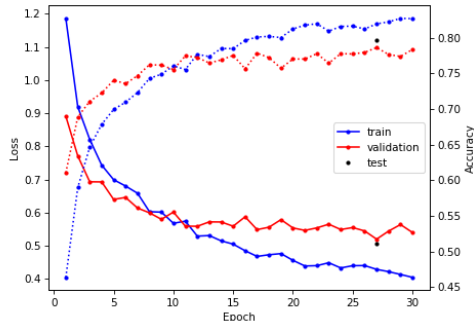
Цитата из статьи

We use MFCCs of dimension 20 with a frame-length of 25ms, mean-normalized over a sliding window of three seconds. We then process the MFCCs features through a pretrained X-Vector network to obtain a high quality voice embedding of fixed dimension 128 [...].

- У данной нейросети есть только 512-мерные выходы.
- При обучении нейросети в `kaldi` выходы нейросети подвергаются обработке — вычитается среднее по всем дикторам и понижается размерность с помощью LDA.
- Мои попытки выполнять подобную постобработку с помощью `sklearn` / `numpy` заканчивались неудачей — при обучении `guesser` отсутствовала генерализация.
- **Итог:** я использовал 512-мерные эмбединги, получаемые напрямую с выходов нейросети.

Обучение Guesser — первые результаты

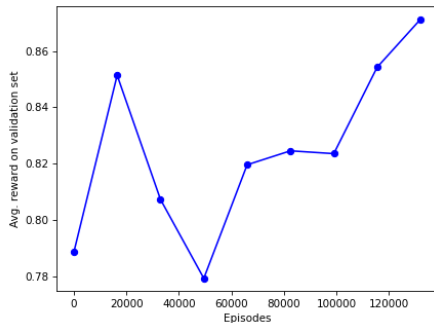
$K = 5$ дикторов и $T = 3$ слова



Увеличение размерности эмбедингов d со 128 до 512 позволило существенно улучшить точность ($\sim 80\%$ вместо 74.1% в оригинальной статье).

Обучение Enquirer — первые результаты

$K = 5$ дикторов и $T = 3$ слова



- Точность ниже, чем в статье (88.6%), но явно есть возможность её улучшить.
- Обучение с помощью PPO, награда $R = 1$ выдается в том случае, когда в конце эпизода guesser правильно угадывает диктора.

Обучение Guesser — подбор гиперпараметров

$K = 5$ дикторов и $T = 3$ слова

Средняя награда (точность) на валидационной выборке:

		learning rate			
		1.00E-03	5.00E-04	1.00E-04	5.00E-04
batch size	20	73.6	78.0	89.0	88.3
	40	78.7	87.0	91.7	90.3
	80	80.6	89.8	93.0	92.4
	160	81.7	91.0	92.7	92.5

- Такая же точность достигается и на тестовой выборке.
- Обучение enquirer позволяет увеличить точность до 96%.

Дальнейшие планы

- Оптимизация гиперпараметров и доработка enquirer.
- Реализация эвристических агентов и сравнение их с нейросетевой моделью.
- Проведение экспериментов в более тяжёлых режимах (больше дикторов, меньше слов).
- Использование других эмбедингов.
- Исследование робастности модели.
- Эксперименты с архитектурой enquirer / guesser.