ЦРТ

РАСПОЗНАВАНИЕ РЕЧИ. ЛЕКЦИЯ 1.

Кореневский Максим Львович, старший научный сотрудник

ВВЕДЕНИЕ В РАСПОЗНАВАНИЕ РЕЧИ

ТЕМЫ ЛЕКЦИИ

- Задачи распознавания речи
- Приложения систем распознавания речи
- Оценка качества/сравнение систем распознавания
- Программное обеспечение для построения систем распознавания
- Классическая структура системы распознавания речи
- Трудности при создании систем распознавания речи
- Простейшая система распознавания на основе сравнения с эталонами

ЗАДАЧИ РАСПОЗНАВАНИЯ РЕЧИ

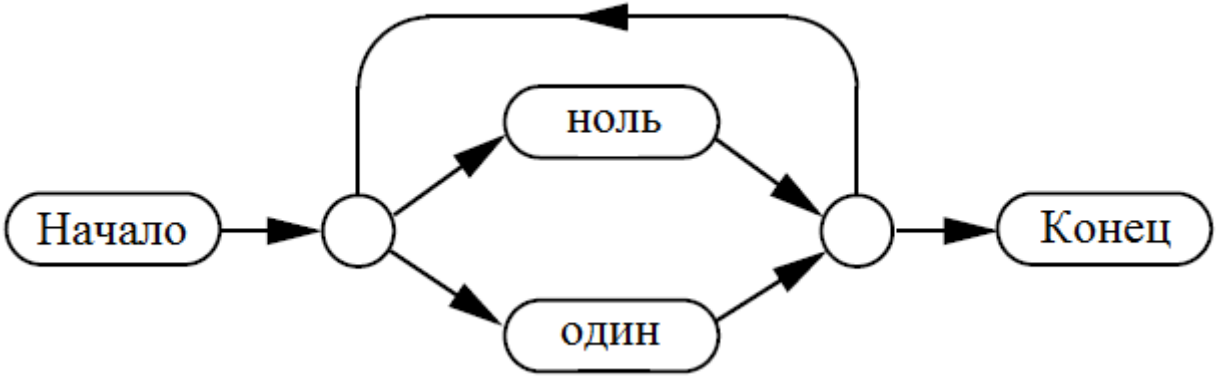
Распознавание фиксированного набора слов/фраз

Слово/фраза	Оценка (score)
Здравствуйте!	30
До свидания	5
Как тебя зовут?	95
Меню, пожалуйста	10
…	

ЗАДАЧИ РАСПОЗНАВАНИЯ РЕЧИ

Распознавание по грамматике

- Грамматика определяет допустимые последовательности слов
- В грамматике могут быть ветвления и циклы (петли)
- Можно сопоставлять определенным путям конкретные действия
- Существуют стандарты описания грамматик (например SRGS)



ЗАДАЧИ РАСПОЗНАВАНИЯ РЕЧИ

Распознавание слитной речи

- Не накладывается никаких ограничений на последовательность слов
- Приходится учитывать ограничения, существующие в самом языке
- Большой размер «словаря распознавания»

Однажды в студеную зимнюю пору….
Однажды в студеную зимнюю гору…
Однажды в студеную зиму набору…
Однажды в суденышке мимо забора…
…..

ЗАДАЧИ РАСПОЗНАВАНИЯ РЕЧИ

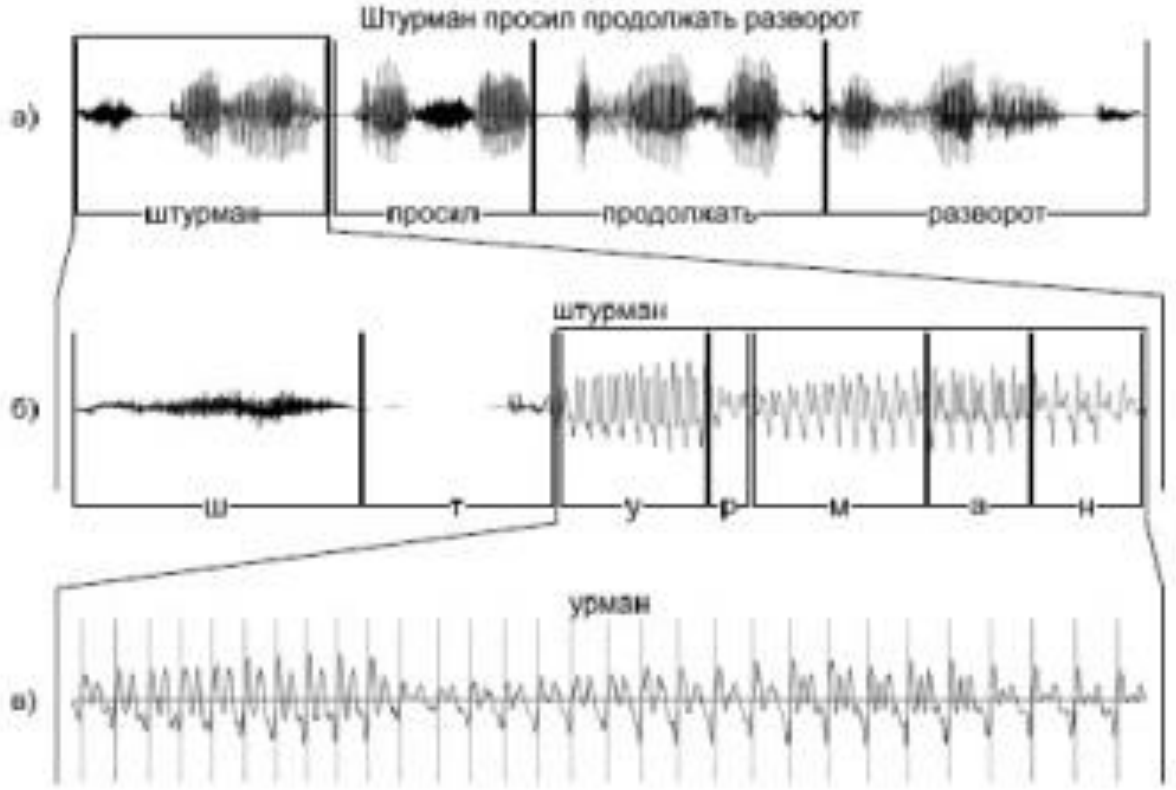
Поиск ключевых слов

– Я хочу купить билеты из Москвы в Санкт-Петербург на Сапсан

Билеты: с 1.13 с. по 1.78 с., уверенность 0.93
Поезд: с 2.22 с. по 2.80 с., уверенность 0.13
Вокзал: с 3.40 с. по 3.96 с., уверенность 0.32
Сапсан: с 3.72 с. по 4.38 с., уверенность 0.98
….

ЗАДАЧИ РАСПОЗНАВАНИЯ РЕЧИ

Сегментация речи



ВВЕДЕНИЕ В РАСПОЗНАВАНИЕ РЕЧИ

ТЕМЫ ЛЕКЦИИ

- Задачи распознавания речи
- Приложения систем распознавания речи
- Оценка качества/сравнение систем распознавания
- Программное обеспечение для построения систем распознавания
- Классическая структура системы распознавания речи
- Трудности при создании систем распознавания речи
- Простейшая система распознавания на основе сравнения с эталонами

ПРИЛОЖЕНИЯ СИСТЕМ РАСПОЗНАВАНИЯ РЕЧИ

Диктовка:

- документы
- электронные письма
- заметки и т.д.

ПРИЛОЖЕНИЯ СИСТЕМ РАСПОЗНАВАНИЯ РЕЧИ

Расшифровка:

- стенограммы
- лекции
- телефонные переговоры

ПРИЛОЖЕНИЯ СИСТЕМ РАСПОЗНАВАНИЯ РЕЧИ

Системы поиска ключевых слов:

- Акустический поиск
  - Малый словарь, работает online
  - Приложения: следственные действия, борьба с терроризмом, голосовое управление, системы «умный дом», контроль качества обслуживания в офисах продаж
- Индексированный поиск
  - Произвольный словарь, работает с большими корпусами речевых данных, создает «индекс» для быстрого поиска
  - Приложения: поиск в базах речевых документов (фильмы, переговоры, лекции и т.д.)

ПРИЛОЖЕНИЯ СИСТЕМ РАСПОЗНАВАНИЯ РЕЧИ

Распознавание по грамматикам:

- IVR-системы
  - Банки
  - Контакт-центры
  - Киоски голосового самообслуживания
- Контроль переговоров, соблюдение регламента
  - Произвольный словарь, работает с большими корпусами речевых данных, создает «индекс» для быстрого поиска
  - Приложения: поиск в базах речевых документов (фильмы, переговоры, лекции и т.д.)

ПРИЛОЖЕНИЯ СИСТЕМ РАСПОЗНАВАНИЯ РЕЧИ

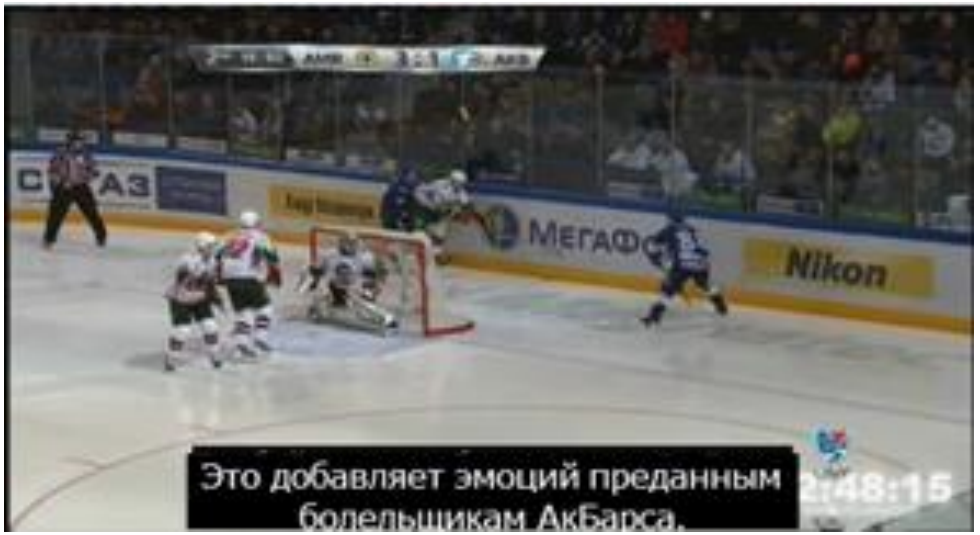
NLU-системы:

- Классификация речевых сообщений по тематике
- Извлечение смысла речевого сообщения
- Диалоговые системы
- Голосовые помощники (Apple Siri, Microsoft Cortana, Amazon Echo и т.п.)
- и т.д.

ПРИЛОЖЕНИЯ СИСТЕМ РАСПОЗНАВАНИЯ РЕЧИ

Прочие приложения:

- Автоматическая подготовка субтитров
- Разметка и аннотирование медиа-баз



ВВЕДЕНИЕ В РАСПОЗНАВАНИЕ РЕЧИ

ТЕМЫ ЛЕКЦИИ

- Задачи распознавания речи
- Приложения систем распознавания речи
- Оценка качества/сравнение систем распознавания
- Программное обеспечение для построения систем распознавания
- Классическая структура системы распознавания речи
- Трудности при создании систем распознавания речи
- Простейшая система распознавания на основе сравнения с эталонами

ОЦЕНКА КАЧЕСТВА/СРАВНЕНИЕ СИСТЕМ РАСПОЗНАВАНИЯ

Распознавание по грамматикам:

- Оценивается точное распознавание всей фразы/последовательности слов
- Естественная мера качества: SER (string error rate) – доля верно распознанных фраз
- Вычисляется в процентах:

$$SER = \frac{\#\text{неверно распознанных фраз}}{\#\text{распознаваемых фраз}} * 100\%$$

ОЦЕНКА КАЧЕСТВА/СРАВНЕНИЕ СИСТЕМ РАСПОЗНАВАНИЯ

Распознавание слитной речи:

- Выравнивание (по Левенштейну):
  - Эталон: Мой дядя, самых честных правил, когда не в шутку занемог…
  - Распознано: Мой дятел самых честь не правил когда в шутку за не мог
  - Замены (substitutions)
  - Вставки (insertions)
  - Удаления (deletions)
- WER (word error rate) – пословная ошибка распознавания
- Accuracy – точность распознавания

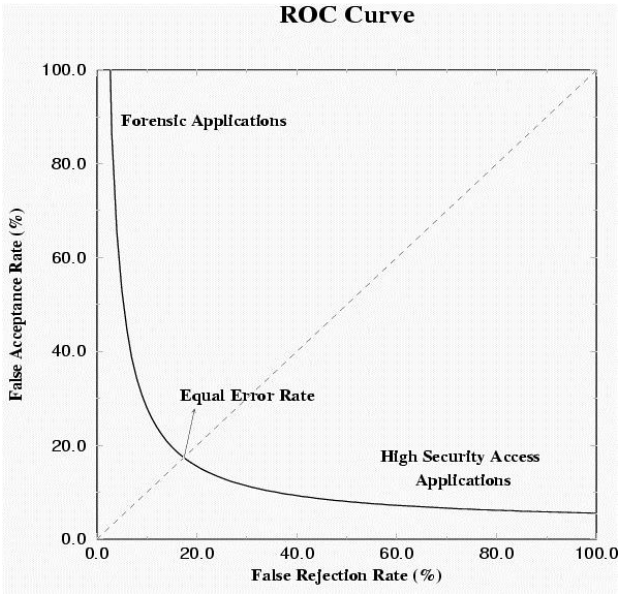
$$WER = \frac{\#\text{замен} + \#\text{вставок} + \#\text{удалений}}{\#\text{слов в эталоне}} * 100\%$$

$$Accuraccy = 100\% - WER$$

ОЦЕНКА КАЧЕСТВА/СРАВНЕНИЕ СИСТЕМ РАСПОЗНАВАНИЯ

Поиск ключевых слов:

- Метрики FR(false rejection) и FA (false acceptance/false alarm)
  - Эталон: Мой дядя, самых честных правил, когда не в шутку занемог…
  - Слова для поиска: дядя, тетя, когда, утка
  - Найдено: дядя, утка; Верно найдено: дядя
  - Ложный пропуск (false rejection): когда
  - Ложное срабатывание (false acceptance): утка
  - $FR = \frac{\#\text{ ложных пропусков}}{\#\text{ ключевых слов во фразе}} * 100\%,$
  - $FA = \frac{\#\text{ ложных срабатываний}}{\#\text{ НЕключевых слов во фразе}} * 100\%.$



- ROC-кривая: зависимость FA от FR при различных настройках системы
- TWV (term weighted value): суммарная мера надежности поиска

$$TWV = 100\% - (FR + \beta * FA), \qquad \beta \approx 1000$$

ВВЕДЕНИЕ В РАСПОЗНАВАНИЕ РЕЧИ

ТЕМЫ ЛЕКЦИИ

- Задачи распознавания речи
- Приложения систем распознавания речи
- Оценка качества/сравнение систем распознавания
- Программное обеспечение для построения систем распознавания
- Классическая структура системы распознавания речи
- Трудности при создании систем распознавания речи
- Простейшая система распознавания на основе сравнения с эталонами

ПО ДЛЯ ПОСТРОЕНИЯ СИСТЕМ РАСПОЗНАВАНИЯ

«Старые» системы:

- HTK (HMM ToolKit). Разработана в Кембридже (UK). Начало разработки – 1989 год.
  - Набор утилит командной строки с единообразным интерфейсом. Написан на C/C++
  - На момент выхода поддерживала большинство современных технологий
  - Активно развивалась до начала 2010-х
  - Проприетарная, но можно получить код.
- CMU Sphinx. Разработана в CMU (Carnegie-Mellon University, USA). Первые релизы – 2000
  - Современные версии написаны на Java, поэтому высоко кросс-платформенные
  - Распространяется под лицензией BSD (open source)
  - Pocket Sphinx – версия для мобильных устройств
- Julius. Разработана в технологическом институте Nagoya, Япония. Начало разработки - 1997 год
- RWTH ASR (RASR). Разработана в университете RWTH, в Aachen, Германия.

ПО ДЛЯ ПОСТРОЕНИЯ СИСТЕМ РАСПОЗНАВАНИЯ

KALDI speech recognition toolkit:

- Первый разработчик: Daniel (Dan) Povey. Первый релиз – 2011 год.
- Open-source, лицензия Apache 2.0, код написан на высокоуровневом C++
- Набор утилит командной строки (ориентация на Linux-подобные системы), bash-скрипты
- Поддержка большинства современных технологий распознавания, быстрое внедрение новых
- Расширенная поддержка использования библиотек линейной алгебры (BLAS, LAPACK)
- Поддержка вычислений на GPU-ускорителях (CUDA)
- Большое количество «рецептов» для построения самых разнообразных систем распознавания.
- Де-факто основной стандарт для исследователей в распознавании речи.

ВВЕДЕНИЕ В РАСПОЗНАВАНИЕ РЕЧИ

ТЕМЫ ЛЕКЦИИ

- Задачи распознавания речи
- Приложения систем распознавания речи
- Оценка качества/сравнение систем распознавания
- Программное обеспечение для построения систем распознавания
- Классическая структура системы распознавания речи
- Трудности при создании систем распознавания речи
- Простейшая система распознавания на основе сравнения с эталонами

СТРУКТУРА СИСТЕМЫ РАСПОЗНАВАНИЯ РЕЧИ

ВВЕДЕНИЕ В РАСПОЗНАВАНИЕ РЕЧИ

ТЕМЫ ЛЕКЦИИ

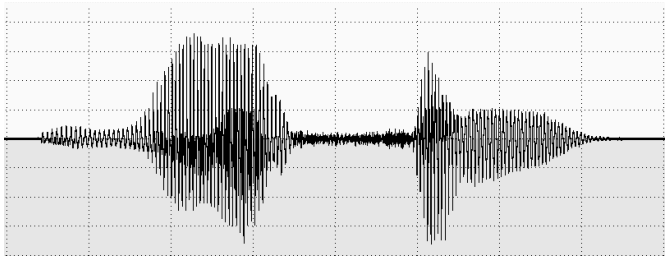
- Задачи распознавания речи
- Приложения систем распознавания речи
- Оценка качества/сравнение систем распознавания
- Программное обеспечение для построения систем распознавания
- Классическая структура системы распознавания речи
- Трудности при создании систем распознавания речи
- Простейшая система распознавания на основе сравнения с эталонами

ТРУДНОСТИ СОЗДАНИЯ СИСТЕМ РАСПОЗНАВАНИЯ

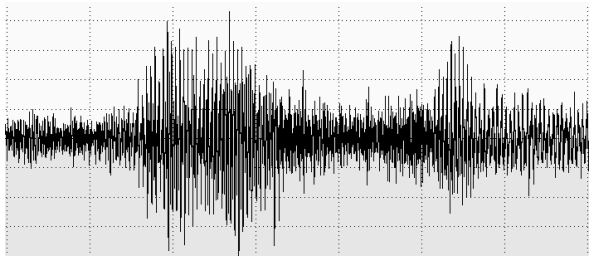
Разнообразие условий записи:

- Окружающие шумы и помехи
- Влияние канала и среды передачи звука (микрофон, стационарный/мобильный телефон)
- Реверберация (переотражения от стен помещения и предметов)
- Частота дискретизации (8000, 11025, 16000, 22050, 44100 Гц)
- Квантование и кодирование
- Клиппирование

Чистая речь («восемь»)



С шумом кафе (SNR=0dB)



ТРУДНОСТИ СОЗДАНИЯ СИСТЕМ РАСПОЗНАВАНИЯ

Междикторская и внутридикторская вариативность:

- Разнообразие голосов (пол, возраст, социальное положение, образование)
- Различные региональные акценты («оканье» и т.п.)
- Дефекты речи (картавость, шепелявость и т.д.)
- Эмоциональное состояние (безразличие, гнев, радость, возбуждение …)
- Физическое состояние (усталость, простуженность/охриплость …)

ТРУДНОСТИ СОЗДАНИЯ СИСТЕМ РАСПОЗНАВАНИЯ

Разнообразие стилей речи:

- Подготовленная (продуманная) речь
- Чтение текста
- Спонтанная речь
  - Различный темп
  - «Проглатывание» окончаний слов
  - Повторы слов, куски слов, «само-исправления»
  - Паузы хезитации («ээээ», «мм»)
  - Слова-паразиты и междометия, нарушающие естественный порядок слов

ТРУДНОСТИ СОЗДАНИЯ СИСТЕМ РАСПОЗНАВАНИЯ

Размеры словаря:

- Распознавание последовательностей цифр – 10 слов
- Распознавание имен и фамилий – сотни слов
- Распознавание новостей – тысячи слов
- Распознавание общей лексики – сотни тысяч слов
- Размер «эффективного» словаря зависит от языка:
  - Для английского языка 99% текстов покрываются 65 тыс. слов
  - Для русского языка 99% текстов покрываются ~500 тыс. **словоформ**
- Размер словаря растет из-за:
  - Богатой морфологии
  - Флективности (изменение окончаний по падежам/родам/числам)
  - Агглютинативности (добавление разных префиксов/аффиксов уточняет значение)
  - и т.д.

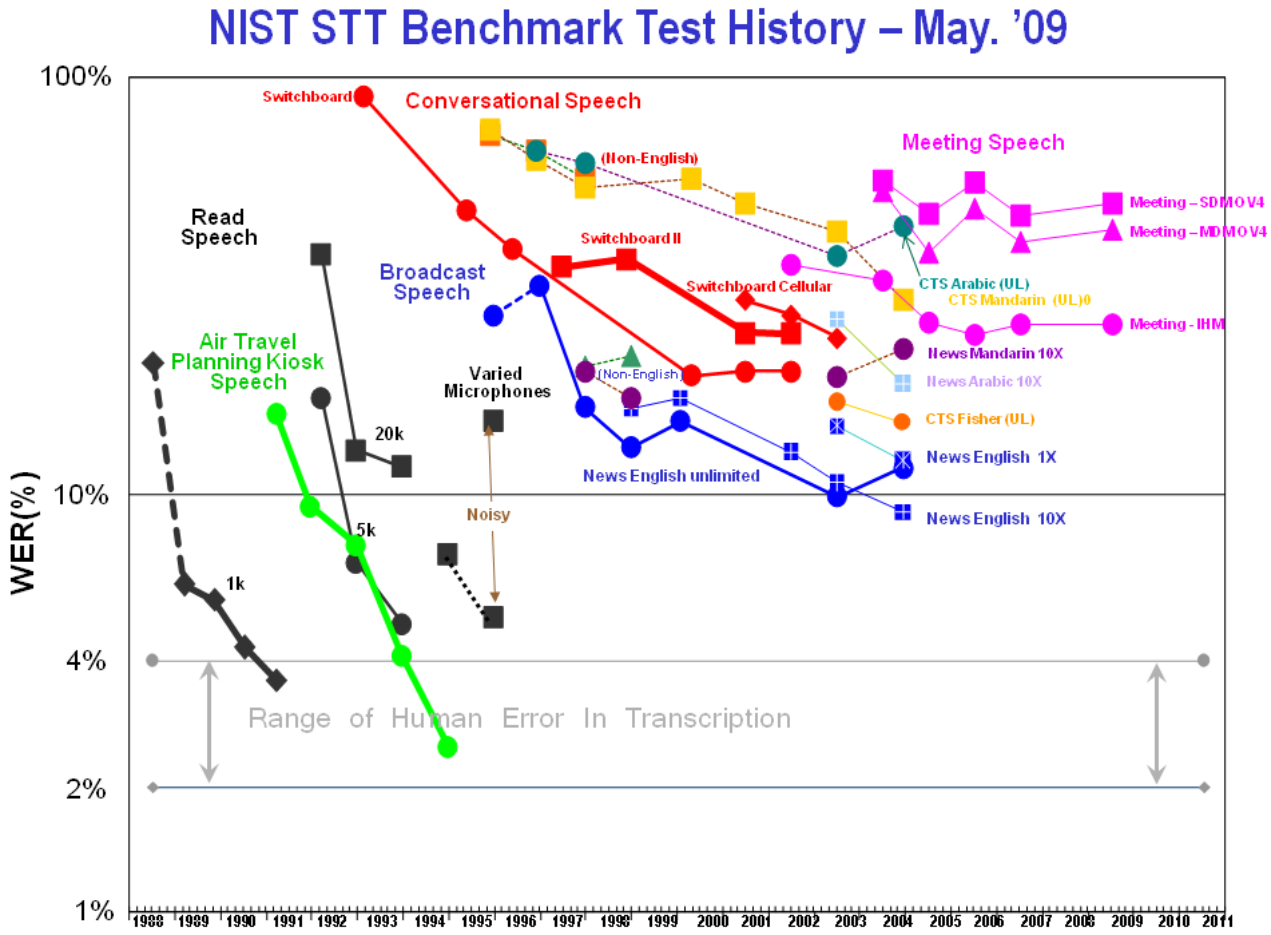
ТРУДНОСТИ СОЗДАНИЯ СИСТЕМ РАСПОЗНАВАНИЯ

Сбор и подготовка данных для обучения:

- Подготовка акустической базы (для обучения акустической модели)
  - Запись фонограмм
  - Предобработка фонограмм (разделение каналов стерео, шумоочистка, нарезка)
  - Подготовка эталонных текстовок
  - Аннотирование речевых данных (диктор, канал записи, особенности записи и т.д.)
- Подготовка текстовой базы (для обучения языковой модели)
  - Набор текстовых данных из различных источников (книги, фильмы, телефонные разговоры, социальные сети, Википедия, TV-программы, выпуски новостей и т.д.)
  - Парсинг и фильтрация данных (удаление html-тегов, повторов, рекламы и т.д.)
  - Нормализация текстов (регистр, кодировка, раскрытие числительных, аббревиатур и т.д.)

ТРУДНОСТИ СОЗДАНИЯ СИСТЕМ РАСПОЗНАВАНИЯ

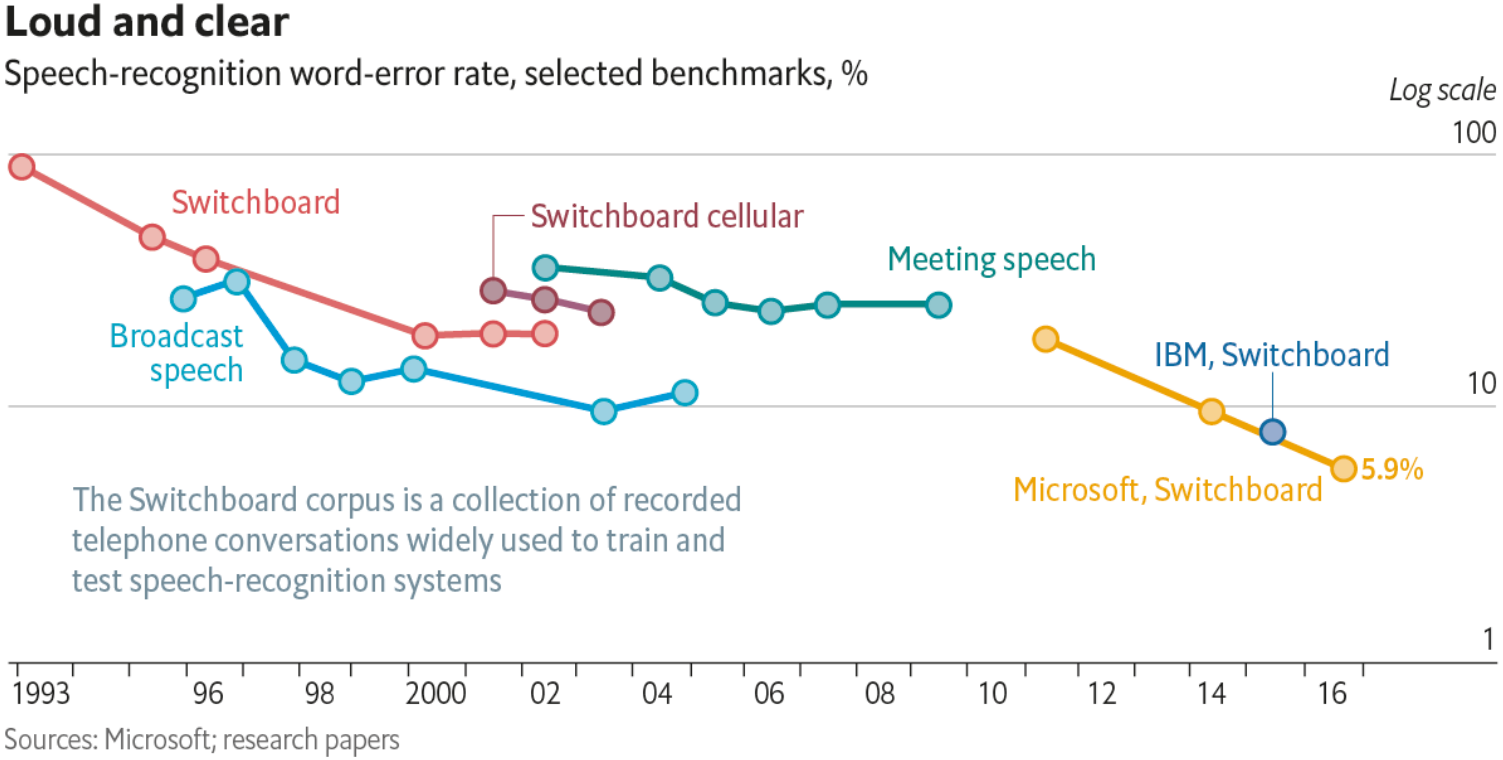
Показатели state-of-the-art систем распознавания



ТРУДНОСТИ СОЗДАНИЯ СИСТЕМ РАСПОЗНАВАНИЯ

Показатели state-of-the-art систем распознавания

The Economist, May-2017



ВВЕДЕНИЕ В РАСПОЗНАВАНИЕ РЕЧИ

ТЕМЫ ЛЕКЦИИ

- Задачи распознавания речи
- Приложения систем распознавания речи
- Оценка качества/сравнение систем распознавания
- Программное обеспечение для построения систем распознавания
- Классическая структура системы распознавания речи
- Трудности при создании систем распознавания речи
- Простейшая система распознавания на основе сравнения с эталонами

СРАВНЕНИЕ С ЭТАЛОНАМИ

Постановка задачи. Общая идея.

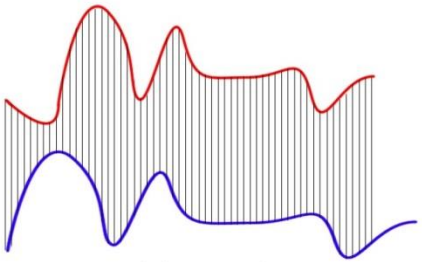
- Хочется распознавать небольшое количество фиксированных слов/фраз
- Пользователь записывает по несколько экземпляров каждого слов
- В test-time система **«сравнивает»** записанный звук с каждым из эталонов
- Слово, соответствующее ближайшему эталону, – результат распознавания!
- Главный вопрос: а как сравнивать две фонограммы?
  - Вычислим признаки для каждой из фонограмм (например MFCC – 13 мерные векторы)
  - Векторы можно сравнивать друг с другом, например с помощью Евклидова расстояния

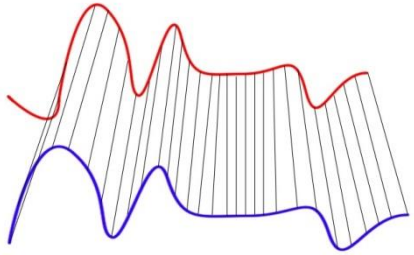
$$d(X,Y) = \sqrt{\sum_i (x_i - y_i)^2}$$

  - Но как сравнить две последовательности векторов разной длины?

Алгоритм DTW (dynamic time warping)

- Идея: «деформировать» шкалу времени для каждой из фонограмм, так чтобы минимизировать суммарное отклонение признаков двух фонограмм


Euclidean Matching


Dynamic Time Warping Matching

- Формально: найти такие последовательности индексов $1 = i_1 \leq \cdots \leq i_N = T_1$ и $1 = j_1 \leq \cdots \leq j_N = T_2$ (выравнивание), что

$$D = \sum_{k=1}^{N} d(X^{i_k}, Y^{j_k}) \to \min$$

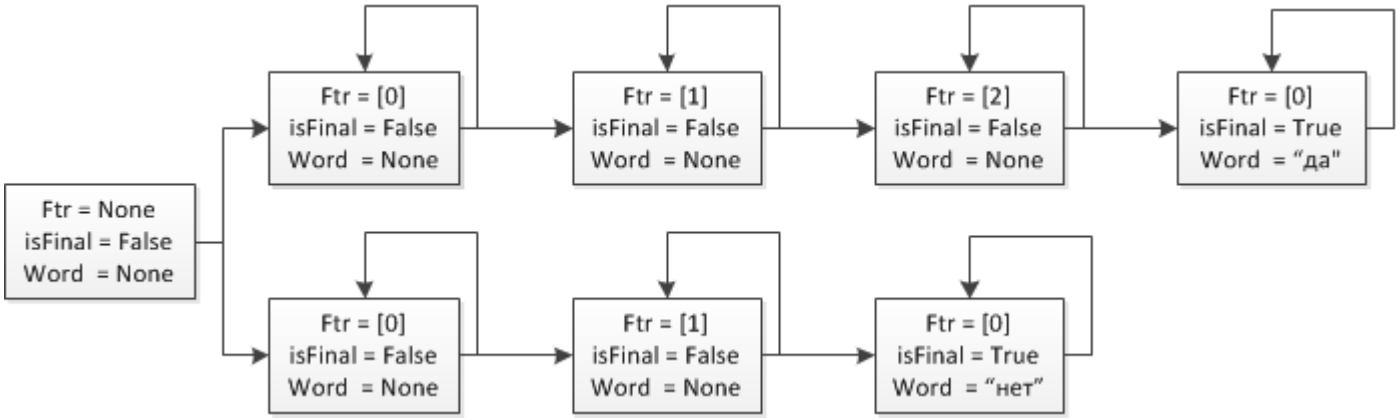
Алгоритм DTW (продолжение)

- Алгоритм динамического программирования (Bellman).
- Введем промежуточную функцию $D(i,j)$ - расстояние от первых i кадров последовательности X до первых j кадров последовательности Y.
- Для нее справедлив простой рекурсивный способ вычисления:
  - $D(1,1) = d(X^1, Y^1)$
  - $D(1,j) = D(1,j-1) + d(X^1, Y^j),\ \ j > 1, \qquad D(i,1) = D(i-1,1) + d(X^i, Y^1),\ \ i > 1$
  - $D(i,j) = \min(D(i-1,j), D(i,j-1), D(i-1,j-1)) + d(X^i, Y^j),\quad i,j > 1$
  - $D = D(T_1, T_2)$.
- Иллюстрация:

	-2	10	-10	15	-13	20	-5	14	2
3	5	12	25	37	53	70	78	89	90
-13	16	28	15	43	37	70	78	105	104
14	32	20	39	16	43	43	62	62	74
-7	37	37	23	38	22	49	45	66	71
9	48	38	42	29	44	33	47	50	57
-2	48	50	46	46	40	55	36	52	54

Token-passing алгоритм

- Построение (направленного) графа распознавания:
  - Каждый кадр (вектор признаков) каждого эталона свяжем с одним состоянием графа
  - Дополним состояние меткой конца фразы (isFinal), словом, соответствующим эталону (только в последнем кадре) и списком следующих состояний.
  - Введем фиктивный стартовый узел (просто для удобства)
- Пример: 2 эталона, «да» (0, 1, 2, 0) и «нет» (0, 1, 0)



Token-passing алгоритм (продолжение)

- Есть тестовая фонограмма – надо найти ближайший эталон
- Токен – структура, связанная с состоянием графа и хранящее текущее расстояние, накопленное при проходе по эталону до этого состояния.
- В каждый момент храним много токенов, при переходе на следующий кадр данный токен либо остаются в том же состоянии, либо перемещается в следующее и обновляет накопленную дистанцию
- В данном состоянии имеет смысл хранить только токен с лучшей дистанцией! (принцип динамического программирования)
- Когда дошли до конца – сравниваем токены в финальных состояниях каждого эталона: токен с наилучшей дистанцией определяет «выигравшее» слово.

Token-passing алгоритм. Псевдокод

```
10 создаём стартовый токен, помещаем его в виртуальный узел, помещаем в activeTokens
20 for frame in РаспознаваемыйФайл:
30        for token in activeTokens:
40                  for переход in всеВозможныеПереходыИз(token.state):
50                            newToken=создать новый токен в узле, куда указывает переход
50                            скопировать всё из token в newToken
60                            newToken.dist+=расст(frame, КадрЭталонаТамКудаМыПерешли)
70                            nextTokens.append(newToken)
80        закончили обработку кадра
90        проредить токены, оставив в каждом узле графа только токен с лучшей дистанцией
100       activeTokens=nextTokens
110       очистить nextTokens
120 закончили обработку записи
130 Для выдачи результата, перебрать все токены, дошедшие до финальных узлов. Токен с
    лучшим расстоянием победил.
```

Token-passing алгоритм. Достоинства и недостатки

- Плюсы алгоритма:
  - Компактное представление всех эталонов сразу
  - Удобный и единообразный алгоритм обработки
  - Легко обобщается на более сложные графы (рассмотрим в следующих лекциях)
- Слабость описанного варианта Token-passing по сравнению с обычным DTW: он позволяет «сжимать» эталоны, растягивая тестовую фонограмму относительно них, но не позволяет обратного (нельзя пройти сразу несколько состояний в графе за 1 кадр тестовой фонограммы)

СИСТЕМА РАСПОЗНАВАНИЯ: СРАВНЕНИЕ С ЭТАЛОНАМИ

Достоинства и недостатки DTW:

- Плюсы:
  - Интуитивность идеи
  - Простота реализации
  - Допустимо создавать эталоны не слов, а произвольных звуков
- Минусы:
  - Необходимость хранить все эталоны
  - Ограниченность набора эталонов в смысле обобщающей способности
  - Невысокая точность
  - Маленький объем словаря
- Выход:
  - Создавать «модели» слов, описывающие все потенциальное множество их эталонов.
  - Обучать модели можно по большим объемам данных.
  - Для распознавании использовать только сами модели, без эталонов.

ЦРТ

РАСПОЗНАВАНИЕ РЕЧИ. ЛЕКЦИЯ 2.

Кореневский Максим Львович, старший научный сотрудник

СОВРЕМЕННЫЕ СИСТЕМЫ РАСПОЗНАВАНИЯ

ТЕМЫ ЛЕКЦИИ

- Вероятностная постановка задачи распознавания речи
- Марковские цепи
- Скрытые Марковские модели (Hidden Markov Models, HMM)
- Языковые модели
- Графы распознавания и WFST

ПОСТАНОВКА ЗАДАЧИ РАСПОЗНАВАНИЯ

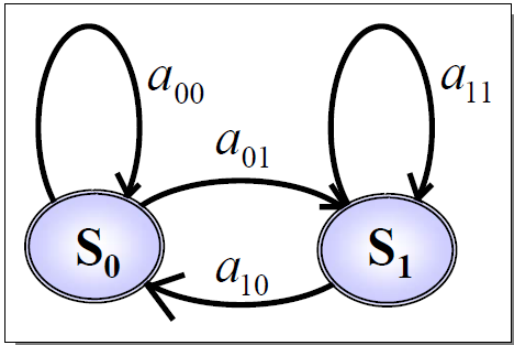
Вероятностная трактовка задачи

- Есть последовательность наблюдений $O = (o_1, o_2, \dots, o_T)$
- Требуется найти такую последовательность слов $W = (w_1, w_2, \dots, w_n)$, которая максимизирует вероятность $P(W|O)$, т.е.

$$W = \arg\max_W P(W|O) = \arg\max_W \frac{p(O|W)P(W)}{p(O)} = \arg\max_W p(O|W)P(W)$$

- $p(O|W)$ - правдоподобие (likelihood) последовательности наблюдений на данной последовательности слов. За него отвечает акустическая модель (классификатор).
- $P(W)$ - априорная вероятность данной последовательности слов в языке. За нее отвечает языковая модель.
- За максимизацию всего произведения отвечает декодер.

Марковский процесс с дискретным временем:



- Есть множество состояний $S = \{S_1, S_2, \dots, S_N\}$
- В каждый момент времени процесс находится в одном из состояний.
- Вероятность перехода процесса из данного состояния в другое определяется только данным состоянием (Марковское свойство)
- Набор вероятностей перехода:

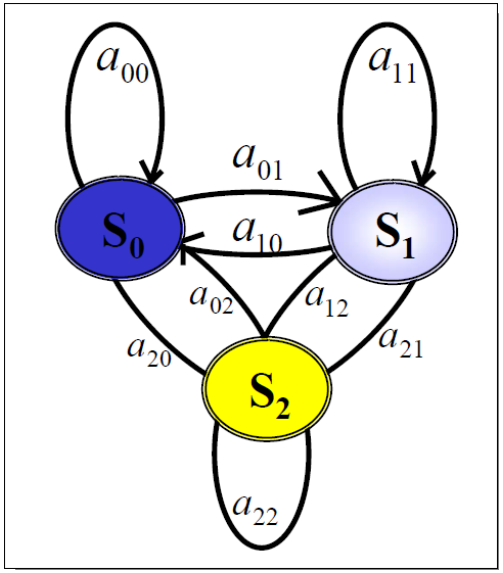
$$a_{ij} = P\left(q_t = S_j \middle| q_{t-1} = S_i\right), \quad t = 1,2, \dots, T,$$

где q_t - состояние, в котором процесс находится в момент времени t.

- Сумма всех вероятностей перехода из одного состояния равна единице

$$\sum_{j=1}^{N} a_{ij} = 1, \qquad i = 1,2, \dots, N$$

Пример: **наблюдаемая** марковская цепь для погоды



- Три состояния:
  - S_1 - дождь
  - S_2 - облака
  - S_3 - солнце
- Вероятности переходов заданы матрицей

$$A = \{a_{ij}\} = \begin{bmatrix} 0.4 & 0.3 & 0.3 \\ 0.2 & 0.6 & 0.2 \\ 0.1 & 0.1 & 0.8 \end{bmatrix}$$

- Как рассчитать вероятность последовательности

  $O = \{\text{солнце}, \text{солнце}, \text{солнце}, \text{дождь}, \text{дождь}, \text{солнце}, \text{облака}, \text{солнце}\}$?
- Ей соответствует последовательность состояний $S = (2, 2, 2, 0, 0, 2, 1, 2)$.
- $P(O|\text{модель}) = P(S = (2, 2, 2, 0, 0, 2, 1, 2)|\text{модель}) =$

  $= P(q_1 = 2)P(q_2 = 2|q_1 = 2) \cdots P(q_8 = 2|q_7 = 1) = \pi_2(0.8)^2 \cdot (0.1) \cdot (0.4) \cdot (0.3) \cdot (0.1) \cdot (0.2).$

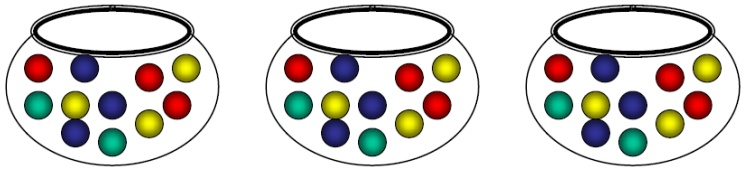
СКРЫТАЯ МАРКОВСКАЯ МОДЕЛЬ

Формальное задание

- Наблюдаемая Марковская модель: $\lambda = (\pi, A)$, где $\pi_i = P(q_1 = i), \quad i = 1,2,\dots,N$ - начальные вероятности
- Скрытая Марковская модель (Hidden Markov Model, HMM): $\lambda = (\pi, A, B)$
  - Состояния процесса больше не наблюдаются непосредственно
  - Есть набор из M наблюдаемых значений $V = \{v_1, v_2, \dots, v_M\}$
  - В каждом состоянии задано вероятностное распределение наблюдаемых в нем значений: $b_j(k) = P(o_t = v_k | q_t = S_j), \quad k = 1,2,\dots,M, \quad t = 1,2,\dots,T$
  - B – набор этих вероятностных распределений: $B = \{b_1, b_2, \dots, b_N\}$

СКРЫТАЯ МАРКОВСКАЯ МОДЕЛЬ

Пример с урнами и шарами



- Есть 3 урны, в каждой из которых определенное известное количество шаров красного, синего, желтого и зеленого цвета. И есть некий «аппарат»:
  - 1. Вначале аппарат выбирает урну наугад в соответствии с некими вероятностями π_i
  - 2. После этого аппарат достает из урны случайно выбранный шар и записывает его цвет
  - 3. Шар возвращается обратно в урну
  - 4. После этого аппарат выбирает, к какой урне переместиться согласно распределению a_{ij}
  - 5. Шаги 2-4 повторяются некоторое количество раз
- Наблюдатель видит только последовательность цветов, записанную аппаратом. Номера урн он не знает! Хочется ответить на вопросы:
  - Какова вероятность выбранной последовательности цветов?
  - Какой последовательности урн она наиболее вероятно соответствует?

СКРЫТАЯ МАРКОВСКАЯ МОДЕЛЬ

HMM как генеративная вероятностная модель

- HMM можно использовать для генерации последовательности наблюдений $O = (o_1, o_2, \ldots, o_T)$:
  1. Положить $t = 1$
  2. Выбрать начальное состояние $q_1 = i$ в соответствии с распределением $\{\pi_i\}$
  3. Сгенерировать наблюдение $o_t = v_k$ в соответствии с распределением $\{b_i(k)\}$
  4. Выбрать следующее состояние $q_{t+1} = j$ в соответствии с распределением $\{a_{ij}\}$
  5. Положить $t = t + 1$ и перейти к шагу 3, если $t \leq T$

СКРЫТАЯ МАРКОВСКАЯ МОДЕЛЬ

1. Вероятность последовательности наблюдений

Требуется вычислить вероятность $\mathrm{P}(O|\lambda)$.

- Рассмотрим все возможные последовательности состояний $q = (q_1, q_2, \dots, q_T)$.
- По формуле полной вероятности $P(O|\lambda) = \sum_q P(O, q|\lambda) = \sum_q P(O|q, \lambda)P(q|\lambda)$.
- Первый сомножитель – вероятность на известной последовательности состояний (предполагаем, что наблюдения независимы при известных состояниях):

$$P(O|q, \lambda) = \prod_{t=1}^{T} P(o_t|q_t, \lambda) = b_{q_1}(o_1) \cdot b_{q_2}(o_2) \cdots b_{q_T}(o_T).$$

- Второй сомножитель – вероятность самой последовательности состояний:

$$P(q|\lambda) = \pi_{q_1} \cdot a_{q_1 q_2} \cdot a_{q_2 q_3} \cdots a_{q_{T-1} q_T} \cdot$$

- Итого

$$P(O|\lambda) = \sum_q \pi_{q_1} b_{q_1}(o_1) a_{q_1 q_2} b_{q_2}(o_2) a_{q_2 q_3} \cdots a_{q_{T-1} q_T} b_{q_T}(o_T).$$

СКРЫТАЯ МАРКОВСКАЯ МОДЕЛЬ

Forward-алгоритм

- Введем вспомогательную величину (forward-вероятность):

$$\alpha_t(i) = P(o_1 o_2 \dots o_t, q_t = i|\lambda)$$

- Для нее справедливы следующие рекурсивные соотношения:

$$\alpha_1(i) = \pi_i b_i(o_1)$$

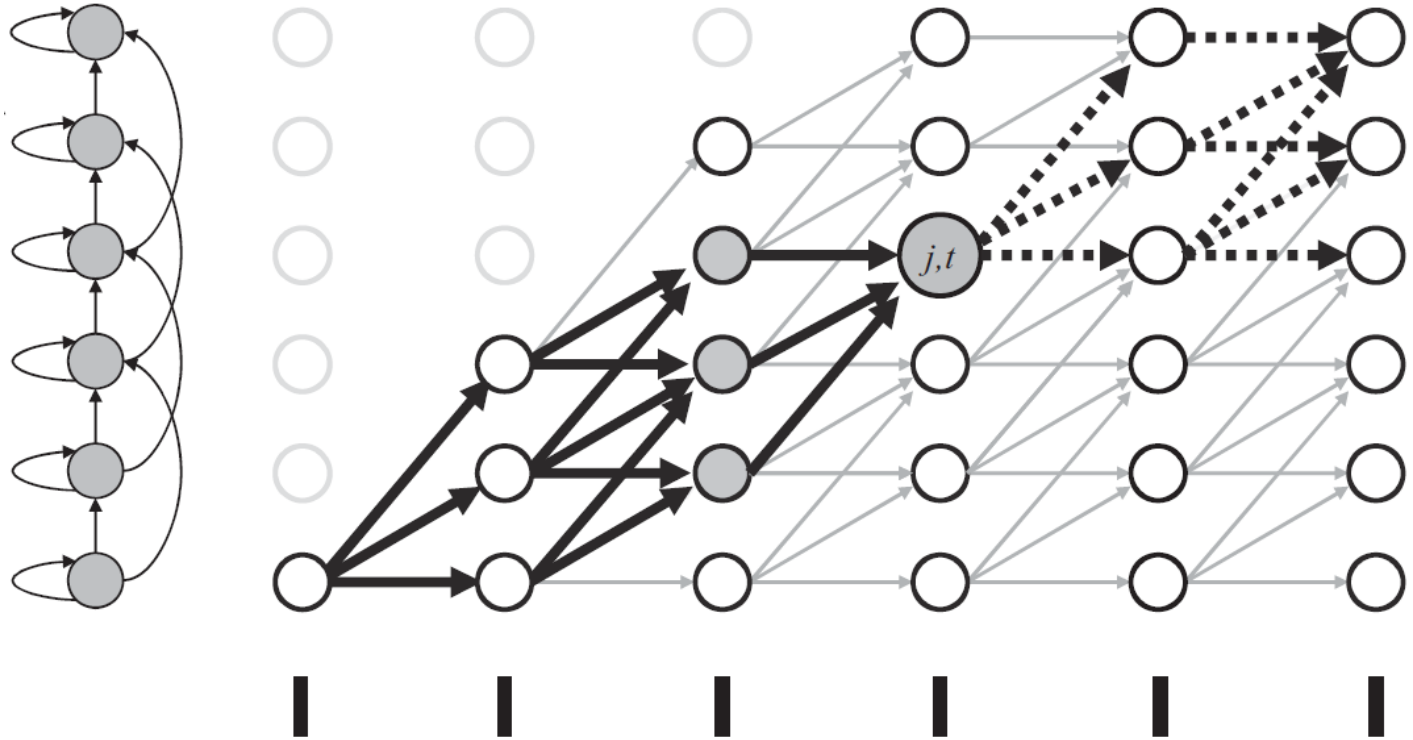
$$\alpha_{t+1}(j) = \left[\sum_{i=1}^{N} \alpha_t(i) a_{ij}\right] b_j(o_{t+1})$$

$$P(O|\lambda) = \sum_{i=1}^{N} \alpha_T(i).$$

- Это позволяет вычислить вероятность за $O(TN^2)$ операций

СКРЫТАЯ МАРКОВСКАЯ МОДЕЛЬ

Forward-алгоритм:



СКРЫТАЯ МАРКОВСКАЯ МОДЕЛЬ

Backward-алгоритм:

- Backward-вероятность:

$$\beta_t(i) = P(o_{t+1}o_{t+2} \dots o_T, q_t = i|\lambda)$$

- Для нее справедливы следующие рекурсивные соотношения:

$$\beta_T(i) = 1, \qquad \beta_t(i) = \sum_{i=1}^{N} a_{ij}\, b_j(o_{t+1})\beta_{t+1}(j), \qquad P(O|\lambda) = \sum_{i=1}^{N} \pi_i b_i(o_1)\beta_0(i).$$

- С помощью forward и backward-вероятностей можно вычислить вероятности прохода через данное состояние в данный момент времени:

$$P(O, q_t = i|\lambda) = P(o_1 o_{t+2} \dots o_T, q_t = i|\lambda) = \alpha_t(i)\beta_t(i)$$

$$P(q_t = i|O, \lambda) = \frac{P(O, q_t = i|\lambda)}{P(O|\lambda)} = \frac{\alpha_t(i)\beta_t(i)}{P(O|\lambda)}$$

2. Вычисление последовательности состояний

- Вторая задача: найти последовательность состояний $q = (q_1, q_2, \dots, q_T)$, которая максимизирует вероятность порождения данной последовательности наблюдений $O = (o_1, o_2, \dots, o_T)$:

$$\hat{q} = \operatorname*{argmax}_q P(O, q|\lambda)$$

- Динамическое программирование: определим вспомогательную величину

$$\delta_t(i) = \max_{q_1 q_2 \dots q_{t-1}} P(q_1 q_2 \dots q_{t-1}, q_t = i, o_1 o_2 \dots o_t|\lambda)\,.$$

- Ее тоже можно вычислять рекурсивно:

$$\delta_1(i) = \pi_i b_i(o_1), \qquad \delta_{t+1}(j) = \max_i \left(\delta_t(i) a_{ij}\right) b_j(o_{t+1})$$

- Если на каждом шаге запоминать из какого состояния $\varphi_t(i)$ мы пришли в данное, то можно восстановить оптимальную последовательность состояний
- Это алгоритм Витерби (A.Viterbi, 1972)

Алгоритм Витерби

- Инициализация: $\delta_1(i) = \pi_i b_i(o_1), \quad \varphi_1(i) = 0, \quad i = 1,2,\dots,N$
- Рекурсия: $\delta_t(j) = \max_i\left(\delta_{t-1}(i)a_{ij}\right) b_j(o_t),$

  $\varphi_t(j) = \operatorname{argmax}_i\left(\delta_{t-1}(i)a_{ij}\right) b_j(o_t), \qquad i = 1,2,\dots,N, \qquad t = 2,\dots,T$
- Завершение: $P^* = \max_i(\delta_T(i))\,,$

  $\hat{q}_T = \operatorname{argmax}_i(\delta_T(i))\,,$
- Обратный ход: $\hat{q}_{t-1} = \varphi_t(\hat{q}_t), \quad t = 2,\dots,T$
- Лучше все вычисления производить в логарифмах (произведения → суммы)

СКРЫТАЯ МАРКОВСКАЯ МОДЕЛЬ

3. Обучение модели

- Пусть наша HMM должна описывать конкретное слово и есть набор «эталонов» этого слова. Как найти наилучшие параметры модели?
- Метод максимального правдоподобия: выбрать такие параметры, для которых достигается максимальная суммарная вероятность на эталонах

$$\hat{\lambda} = \underset{\lambda}{\operatorname{argmax}} P(O|\lambda)$$

- Нет эффективного алгоритма для поиска глобального максимума
- Но есть эффективный итерационный алгоритм, который находит локальный максимум: алгоритм Баума-Уэлша (Baum-Welch algorithm).
- На каждой итерации происходит обновление параметров модели: $\lambda \to \bar{\lambda}$
- Для обновления используются статистики состояний, вычисляемые с помощью forward и backward-вероятностей α и β.

3. Обучение модели по Витерби

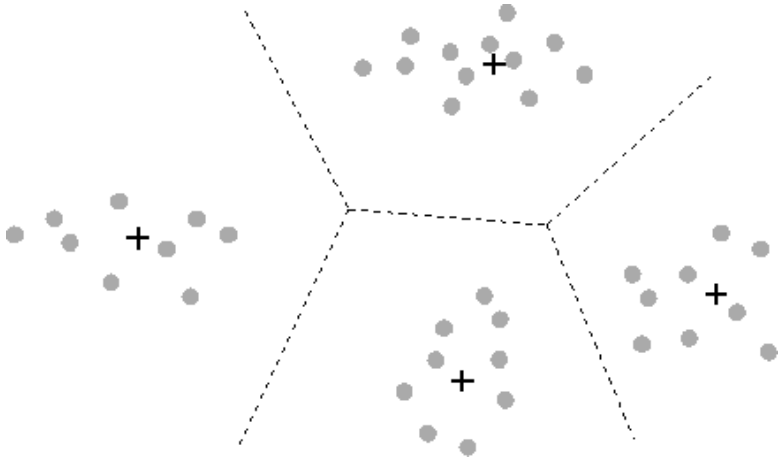
Для обучения можно использовать алгоритм Витерби

1. С текущими значениями параметров провести сегментацию слова на состояния
2. Для каждого состояния i набрать статистику:
   - Сколько раз последовательность состояний начиналась с него
   - Сколько раз в этом состоянии наблюдалось определенное значение v_k
   - Сколько раз из этого состояния был произведен переход в состояние j
3. По статистике начал вычислить новое распределение $\{\pi_i\}$
4. По статистике значений вычислить новые распределения $\{b_i(k)\}$
5. По статистике переходов вычислить новые вероятности переходов $\{a_{ij}\}$
6. Если не сошлось, перейти к шагу 1

СКРЫТАЯ МАРКОВСКАЯ МОДЕЛЬ

Использование непрерывных распределений

- До сих пор мы рассматривали только дискретные распределения в состояниях
- Часто данные распределены непрерывно (например, MFCC)
- Можно провести векторное квантование и привести задачу к дискретной, но это «сжатие с потерями» и оно снижает качество



- Вероятности наблюдений в состояниях заменяются на правдоподобия!
- Как непосредственно использовать непрерывные распределения?

СКРЫТАЯ МАРКОВСКАЯ МОДЕЛЬ

Гауссова смесь (Gaussian mixture model, GMM)

- Это распределение следующего вида:

$$b_j(o_t) = \sum_{k=1}^{M} w_{jk} \mathcal{N}(o_t; \mu_{jk}, \Sigma_{jk}),$$

где веса подчиняются следующему ограничению:

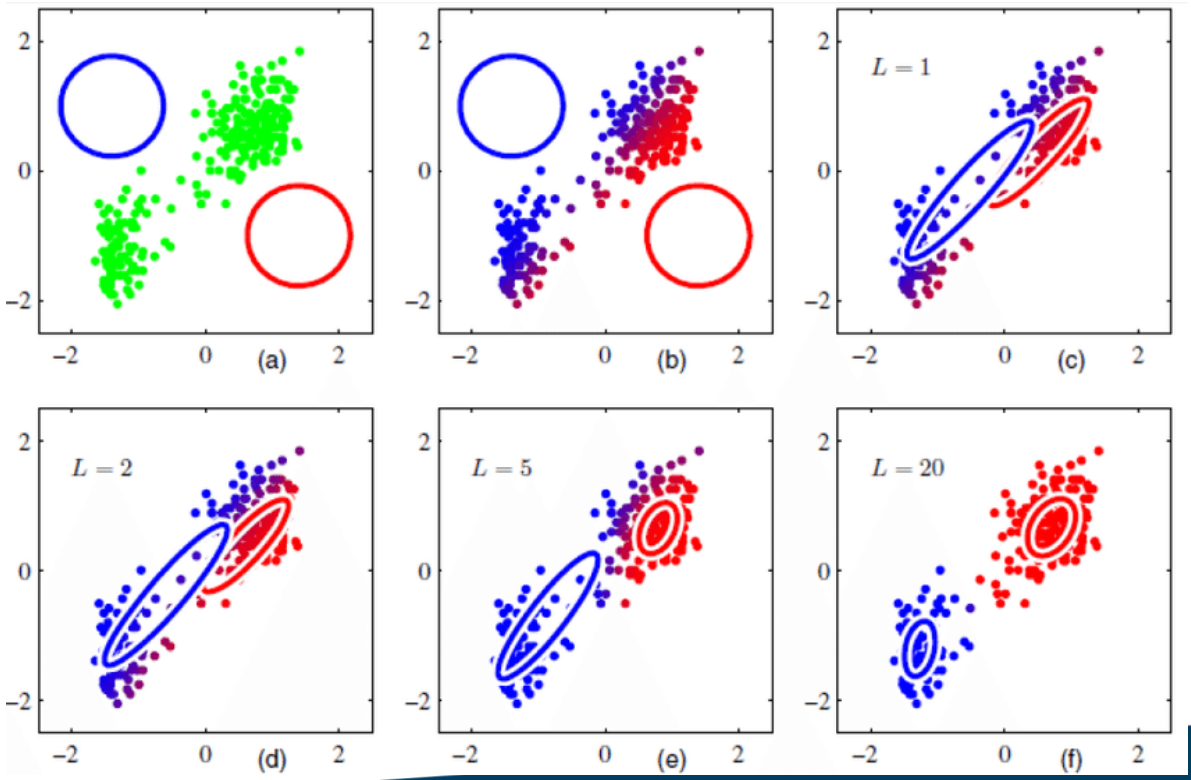
$$w_{jk} \geq 0, \qquad \sum_{k=1}^{M} w_{jk} = 1.$$

- GMM – генеративная модель. Генерировать данные из нее очень просто:
  - Сначала генерируется номер компонента в соответствии с распределением $\{w_{jk}\}$
  - После этого генерируется вектор из распределения $\mathcal{N}(o_t; \mu_{jk}, \Sigma_{jk})$
- С помощью GMM с достаточно большим числом компонентов можно приблизить любое непрерывное распределение с достаточной точностью

СКРЫТАЯ МАРКОВСКАЯ МОДЕЛЬ

Гауссова смесь (Gaussian mixture model, GMM)

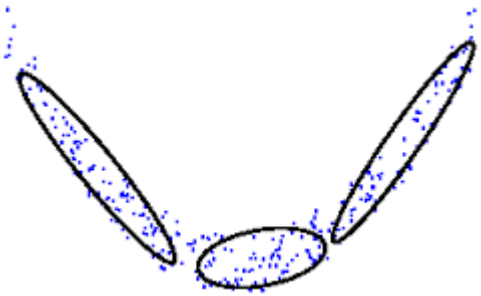
- Для обучения GMM используется метод максимального правдоподобия.
- Применяется итерационный EM-алгоритм (ищет локальный минимум).

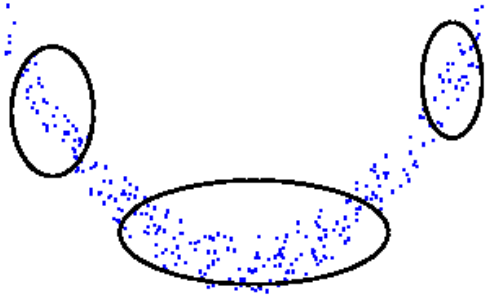


СКРЫТАЯ МАРКОВСКАЯ МОДЕЛЬ

Гауссова смесь (Gaussian mixture model, GMM)

- Существует вариант алгоритма Баума-Уэлша для обучения HMM с GMM
- Схема применения обучения по Витерби вообще не меняется
- GMM с полноковариационными матрицами – очень много параметров. Выход – использование диагональных ковариаций.



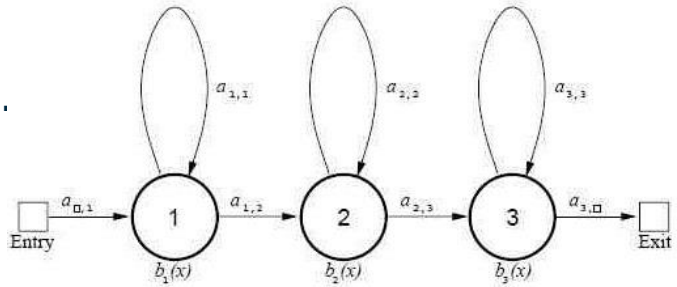


- Декорреляция признаков:
  - Использование MFCC (приближенно некоррелированы благодаря DCT)
  - Использование PCA (Principal Component Analysis), оно же KLT (Karhunen-Loeve Transform)

СКРЫТАЯ МАРКОВСКАЯ МОДЕЛЬ

Недостатки «словных» HMM

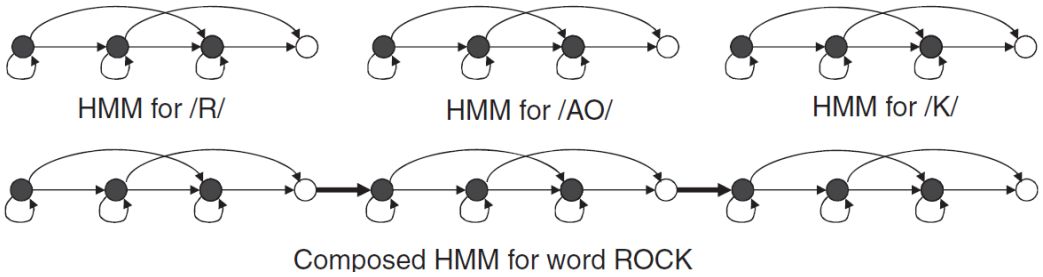
- Много состояний, много параметров («тяжелые» модели)
- С ростом размера словаря становится слишком затратным
- На каждое слово в обучающих данных может быть слишком мало примеров, как следствие – переобучение
- Выход: перейти на суб-словные единицы: слоги, фонемы.
  - Фонем в большинстве языков относительно немного (не более 100)
  - На каждую фонему значительно больше статистики
  - «Топология» HMM для фонем может быть очень простой.
  - Типичный вариант фонемной HMM: 3 state, left-to-right



СКРЫТАЯ МАРКОВСКАЯ МОДЕЛЬ

Фонемные HMM и способы их улучшения

- Как учить:
  - Фонемные HMM «склеиваются» в словные HMM, а они – в HMM для целой фразы



  - Одинаковые состояния одинаковых фонем «разделяют» (share) общие параметры
  - При обучении надо накапливать статистики для состояний по всем вхождениям
- Типичные характеристики:
  - Число состояний – 150-200, число компонентов в GMM – 8-32.
- Основная проблема: в разном окружении фонемы сильно различаются: фонема «т» в слове «вата» совсем не такая, как в слове «строить». Это называется коартикуляцией

СКРЫТАЯ МАРКОВСКАЯ МОДЕЛЬ

Учет контекста

- Фонема в определенном «окружении» называется аллофоном.
- Аллофон с контекстом в 1 фонему слева и справа называется трифоном.
- Фонемная запись: вата => v a0 t a4 (a0 – ударная, a4 – на конце слова)
- Трифонная запись: вата => v+a0 v-a0+t a0-t+a4 t-a4:
  - Трифон v-a0+t – это аллофон фонемы a0, которая находится в окружении v слева и t справа
  - Дифон v+a0 – это аллофон фонемы v справа от которой стоит фонема a0
  - Дифон t-a4 – это аллофон фонемы a4, слева от которой стоит фонема t
- При склеивании словных HMM следует учитывать влияние «межсловного» контекста («город Москва» vs. «город Санкт-Петербург»)
- Используют также «пентафоны» - аллофоны с контекстом 2 фонемы с каждой стороны

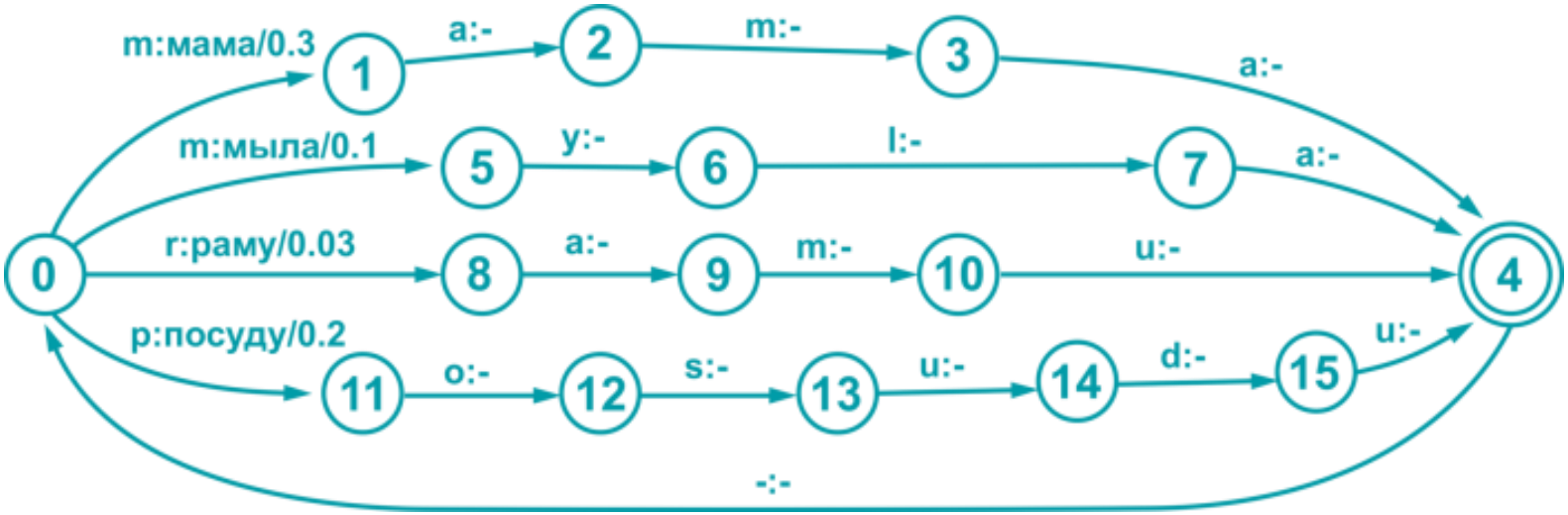
Проблемы трифонных моделей

- Различных трифонов очень много. Если фонем 50, то трифонов 125000.
- Очень многие трифоны из полного набора вообще никогда не встречаются в речи, очень многих в обучающих данных нет вообще или очень мало!
- Чтобы сохранить число параметров в разумных пределах придумали связывать (tie) похожие состояния трифонов. Связанные состояния (сеноны, senones) разделяют (share) общее распределение в состоянии.
- Как правило, связывают трифоны, относящиеся к одной фонеме.
- Связывание обычно проводят по дереву решений (decision tree).
- Дерево строится путем разбиения всего множества трифонов на классы в соответствие с «вопросами».
- Число связанных состояний – обычно 5-10 тысяч.

СКРЫТАЯ МАРКОВСКАЯ МОДЕЛЬ

Распознавание с помощью суб-словных моделей

- Пусть требуется распознавать всевозможные последовательности из слов «мама», «мыла», «раму», «посуду». Строим фонемный граф:



- Этот граф можно преобразовать в трифонный и, далее, в «стейтовый»
- На стейтовом графе можно искать лучший (Витерби) путь с помощью token-passing алгоритма!

СКРЫТАЯ МАРКОВСКАЯ МОДЕЛЬ

Распознавание с помощью суб-словных моделей

- Гипотеза – тот же токен из token-passing алгоритма. Хранит в себе пройденный путь (чтобы можно было восстановить последовательность слов) и накопленное (лог-)правдоподобие.
- На каждом новом кадре
  - Вычисляются правдоподобия (likelihood) всех состояний всех HMM
  - Все активные гипотезы расширяются с учетом возможных переходов из состояния, правдоподобий и вероятностей переходов
  - В каждом состоянии запоминается только лучшая гипотеза (Витерби)
- Если граф большой, то число активных гипотез растет очень быстро. Выход – отсекать (prune) «малоперспективные» гипотезы
- Beam pruning: отсечение гипотез, правдоподобие которых отличается от правдоподобия лучшей более, чем на beam.

Недостатки GMM-HMM моделей

- Динамика ограничена марковским свойством
- Вероятности переходов не зависят от времени
- HMM предполагает, что наблюдения на соседних кадрах независимы и зависят только от состояния, в котором находится модель.
- GMM является «локальной» моделью и учится по ML-критерию
- Для повышения точности надо увеличивать количество компонент GMM, число параметров растет, модель переобучается, расчеты замедляются.

Независимость наблюдений

- Использование дельта-признаков:
  - Пусть имеется последовательность векторов $O = (o_1, o_2, \dots, o_T)$
  - Дополним векторы наблюдений «производными»:

$$\Delta o_t = (o_{t+1} - o_{t-1})/2 \approx \partial o_t / \partial t$$

$$\Delta\Delta o_t = o_{t+1} - 2o_t + o_{t-1} \approx \partial^2 o_t / \partial^2 t$$

- Frame stacking: объединение векторов признаков вокруг текущего в один длинный «супервектор»:

$$O_t = [o_{t-l}^T, \dots, o_{t-1}^T, o_t^T, o_{t+1}^T, \dots, o_{t+r}^T]^T$$

СКРЫТАЯ МАРКОВСКАЯ МОДЕЛЬ

Генеративность и локальность GMM:

- Локальность: пространство оказывается разделено на «области» в каждой из которых ДОМИНИРУЕТ только один компонент GMM, а остальные не оказывают влияние на значения правдоподобия.
- Обучение по критерию ML: параметры модели выбираются так, чтобы максимизировать правдоподобие ЛУЧШЕЙ гипотезы. Но если «конкурирующие» гипотезы близко, то качество распознавания будет низким.
- Дискриминативное обучение: идея – максимально отделить правильную гипотезу от всех конкурирующих. Много различных критериев:
  - MMI (Maximum Mutual Information)
  - MPE (Minimum Phone Error)
  - Maximum Margin и т.д.

СКРЫТАЯ МАРКОВСКАЯ МОДЕЛЬ

Переобучение и усложнение GMM:

- Связывание параметров (не только состояний, но и параметров GMM, отдельных гауссиан, весов, вероятностей переходов)
- Векторное квантование признаков и использование дискретных распределений
- Создание большого пула гауссиан и «набор» отдельных GMM в состояниях из этого пула с различными весами. На этой идее основаны SGMM (subspace GMM)

- Общий вывод: GMM – не самый лучший из возможных классификаторов в состояниях HMM.

Языковая модель:

- Вспомним вероятностную постановку задачи распознавания

$$W = \arg\max_W P(W|O) = \arg\max_W \frac{p(O|W)P(W)}{p(O)} = \arg\max_W p(O|W)P(W)$$

- Стандартное разложение в произведение:

$$P(W) = P(w_1 w_2 \dots w_L) = P(w_1)P(w_2|w_1)P(w_3|w_1 w_2) \cdots P(w_L|w_1 w_2 \dots w_{L-1}) =$$

$$= P(w_1) \prod_{i=2}^{L} P(w_i|w_1 w_2 \dots w_{i-1})$$

- Итак, основная задача - уметь вычислять $P(w_i|w_1 w_2 \dots w_{i-1})$.
- Проблемы:
  - Последовательности могут быть произвольной длины
  - Чем длиннее «история» тем меньше шансов, что такое есть в обучающих данных
  - Некоторые, даже короткие, последовательности вообще никогда не встречаются в языке

ЯЗЫКОВАЯ МОДЕЛЬ

Статистическая n-граммная (n-gram) модель:

- Ограничим максимально возможную длину истории $(n-1)$ словом:

$$P(w_i|w_1 w_2 \dots w_{i-1}) \approx P(w_i|w_{i-n+1} w_{i-n+2} \dots w_{i-1})$$

- Последовательности из n слов $w_{i-n+1} w_{i-n+2} \dots w_{i-1} w_i$ - n-граммы (n-gram):
  - $n = 1$ – униграммы
  - $n = 2$ – биграммы
  - $n = 3$ – триграммы и т.д.
- В результате, например, для триграммной модели:

$$P(\text{"мой дядя самых честных правил"}) =$$
$$= P(\text{"мой"}) \cdot P(\text{"дядя"}|\text{"мой"}) \cdot P(\text{"самых"}|\text{"мой дядя"}) \cdot$$
$$\cdot P(\text{"честных"}|\text{"дядя самых"}) \cdot P(\text{"правил"}|\text{"самых честных"})$$

- В триграммной модели должны присутствовать все униграммные, биграммные и триграммные вероятности.

Оценка вероятностей n-грамм

- Пусть дан большой обучающий текстовый корпус
- Подсчитаем статистику количества вхождений всех n-грамм в нем
- Тогда оценка максимального правдоподобия для вероятности имеет вид:

$$P(w_i|w_{i-n+1}w_{i-n+2}\dots w_{i-1}) \approx \frac{\#\text{ появлений } w_{i-n+1}w_{i-n+2}\dots w_{i-1}w_i}{\#\text{ появлений } w_{i-n+1}w_{i-n+2}\dots w_{i-1}}$$

- Т.е. мы считаем ЧАСТОТУ появления данной последовательности среди всех последовательностей С ТОЙ ЖЕ ИСТОРИЕЙ и разными последними словами в обучающем корпусе
- Обычно в модели сохраняются логарифмы, это позволяет перейти от произведений к суммам и избежать потери точности.

Оценка качества языковой модели:

- Качество ЯМ определяется тем, насколько хорошо она способна предсказывать очередное слово по его истории. Т.е. насколько высокие вероятности она дает РЕАЛЬНЫМ предложениям.
- Перплексия (perplexity) – мера точности ЯМ:

$$PP(w_1 w_2 \dots w_L) = P(w_1 w_2 \dots w_L)^{-1/L} = e^{-\frac{\log P(w_1 w_2 \dots w_L)}{L}}$$

- Т.е. перплексия – это величина, обратная средней (геометрической) вероятности последовательности в расчете на 1 слово
- Перплексия всегда больше единицы, чем меньше перплексия, тем лучше ЯМ
- Важно:
  - Вычислять перплексию на отдельном тексте, не входящем в обучающую выборку
  - Сравнивать разные модели по перплексии на одном и том же тексте (а не на разных)

Discounting

- Для размера словаря в 1000 слов число различных триграмм – миллиард
- НО: большинство триграмм вообще никогда не встречаются в речи
- А часть триграмм в речи есть, но их может не быть в обучающем корпусе (unseen), на них надо бы выделить некоторую долю вероятности.
- Для этого используется discounting:

$$P(w_i|w_{i-n+1}w_{i-n+2}\dots w_{i-1}) \approx \frac{D(\#\text{ появлений } w_{i-n+1}w_{i-n+2}\dots w_{i-1}w_i)}{\#\text{ появлений } w_{i-n+1}w_{i-n+2}\dots w_{i-1}}$$

- $D(C)$ – discount-функция
  - Good-Turing discounting: $D_{GT}(C) = (C+1)\frac{N_{C+1}}{N_c}$,
  - Kneser-Ney discounting: $D_{KN} = \frac{C_{KN}(w_{i-n+1}w_{i-n+2}\dots w_{i-1}w_i)}{\sum_W C_{KN}(w_{i-n+1}w_{i-n+2}\dots w_{i-1}w)}$

Сглаживание, откаты

- Discounting выделяет вероятность под unseen-n-граммы, но не говорит о том, как ее между ними распределить.
- Выход: использовать так-называемые веса «отката» (back-off weights), Katz, 1987:

$$P_{smooth}(w_3|w_1w_2) = \begin{cases} P(w_3|w_1w_2), & \text{если } \#(w_1w_2w_3) > 0 \\ \alpha(w_1w_2)P_{smooth}(w_3|w_2), & \text{если } \#(w_1w_2) > 0 \\ P_{smooth}(w_3|w_2), & \text{в противном случае} \end{cases}$$

- Для вероятностей, входящих в правую часть, используется то же правило:

$$P_{smooth}(w_3|w_2) = \begin{cases} P(w_3|w_2), & \text{если } \#w_2w_3) > 0 \\ \alpha(w_2)P_{smooth}(w_3), & \text{в противном случае} \end{cases}$$

- В классическом arpa-формате ЯМ для всех n-грамм не самого большого порядка указываются $\log P_{smooth}$ и $\log \alpha$.

Декодирование с языковой моделью

- Простейший сценарий:
  - В гипотезах (токенах) хранится пройденный путь.
  - Как только дошли до конца очередного слова запрашиваем у ЯМ его вероятность при данной истории и добавляем ее логарифм в score гипотезы
- Недостатки простейшего сценария:
  - Вероятность гипотезы меняется «скачкообразно»
  - Гипотеза может «выпасть» из beam'а до того, как ее score улучшится благодаря ЯМ
  - Частые обращения к ЯМ, дублирование запросов на похожих гипотезах
- Возможные решения:
  - «Внедрить» языковые вероятности непосредственно в стейтовый граф
  - «Размазать» их по длине слова

Другие типы языковых моделей

- Классовые языковые модели
  - Предсказывают вероятность для «классов слов»
- ЯМ на основе рекуррентных нейронных сетей
  - Обучаются таким образом, чтобы предсказывать текущее слово по окружающим словам
  - Использует перевод пространства слов в непрерывное представление (word embeddings)
- Ансамбли языковых моделей (интерполяция)

Построение графа распознавания

- Для построения стейтового графа надо иметь:
  - Языковую модель или грамматика– показывает возможные переходы из слова в слово (с их вероятностями)
  - Лексикон (словарь транскрипций) – показывает, как произносятся слова, т.е. из каких фонем оно состоит. Может быть несколько транскрипций на слово, причем с разными вероятностями
  - Контекстная информация – какие трифоны получаются из фонем с учетом левого/правого контекстов и связывания состояний
  - Акустическая модель (HMM) – показывает из каких состояний состоит каждый трифон и задает вероятности переходов из состояния в состояние
- К счастью, каждый из этих видов информации можно представить в едином формате – Weighted Finite-State Transducer (WFST). WSFT является вероятностным конечным автоматом, трансформирующим входную последовательность символов в выходную.

Построение графа распознавания

- Традиционно, WFST для отдельных компонентов обозначают так:
  - G – для грамматики или n-граммной ЯМ (переводит слова в предложение)
  - L – для лексикона (переводит фонемы в слова)
  - C – для контекстной зависимости (переводит трифоны в фонемы)
  - H – для HMM (переводит состояния трифонов в сами трифоны)
- WFST можно объединять: операция композиции делает из двух входных WFST такой новый WFST, который переводит вход первого в выход второго
- С помощью композиции можно построить WFST транслирующий последовательность состояний в предложение! $W = H \circ C \circ L \circ G$
- Такой WFST определяет стейтовый граф распознавания для декодера!
- OpenFST – open-source библиотека для работы с WFST

ЦРТ

РАСПОЗНАВАНИЕ РЕЧИ. ЛЕКЦИЯ 3.

Кореневский Максим Львович, старший научный сотрудник

НЕЙРОННЫЕ СЕТИ В РАСПОЗНАВАНИИ РЕЧИ

ТЕМЫ ЛЕКЦИИ

- Использование нейронных сетей в качестве классификаторов
- Гибридные и тандемные системы распознавания
- Deep Neural Networks и их обучение
- Система распознавания на базе HMM
- Другие архитектуры нейронных сетей (сверточные, рекуррентные)
- Альтернативные подходы к обучению нейронных сетей для ASR

НЕЙРОННЫЕ СЕТИ В РАСПОЗНАВАНИИ РЕЧИ

ТЕМЫ ЛЕКЦИИ

- Использование нейронных сетей в качестве классификаторов
- Гибридные и тандемные системы распознавания
- Deep Neural Networks и их обучение
- Система распознавания на базе HMM
- Другие архитектуры нейронных сетей (сверточные, рекуррентные)
- Альтернативные подходы к обучению нейронных сетей для ASR

НЕЙРОННЫЕ СЕТИ В КАЧЕСТВЕ КЛАССИФИКАТОРОВ

Задача распознавания

- Вероятностная постановка задачи:

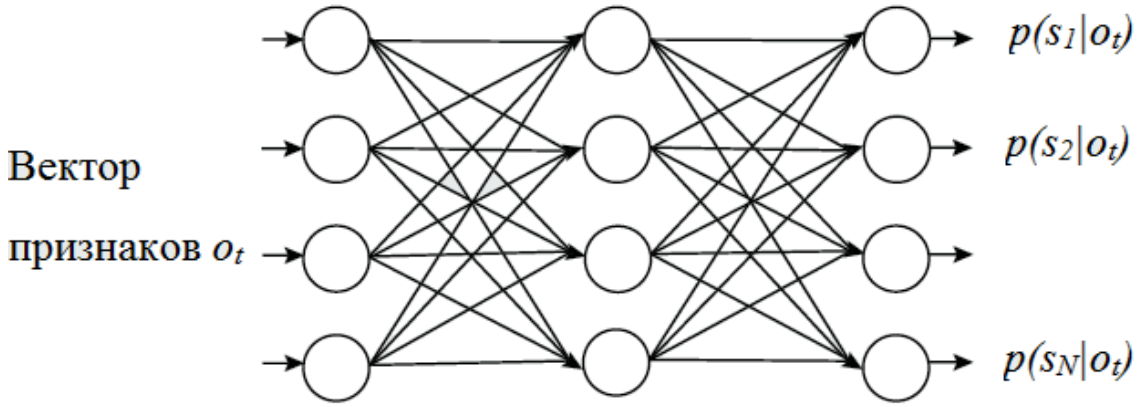
$$W = \arg\max_{W} p(O|W)P(W)$$

- Правдоподобие $p(O|W)$ - вычисляется акустической моделью через вероятности переходов в HMM и правдоподобия в состояниях HMM $b_i(o_t)$.
- Распространенная практика: использование GMM-моделей в качестве классификатора состояний по вектору наблюдений.
- Альтернативный вариант – использовать искусственные нейронные сети (artificial neural network, ANN).
- Нейронная сеть – тоже классификатор, но дискриминативный и не локальный

НЕЙРОННЫЕ СЕТИ В КАЧЕСТВЕ КЛАССИФИКАТОРОВ

Искусственная нейронная сеть

- Как ANN может использоваться в распознавании:



- На выходе нейронной сети – апостериорные вероятности состояний $p(s_i|o_t)$
- А в GMM вычисляются (и используются в декодере) правдоподобия $p(o_t|s_i)$
- Как перевести одно в другое? Формула Байеса: $p(o_t|s_i) = \frac{p(s_i|o_t)}{p(s_i)} p(o_t)$
- Второй сомножитель не зависит от состояний.

НЕЙРОННЫЕ СЕТИ В КАЧЕСТВЕ КЛАССИФИКАТОРОВ

Как обучать ANN для распознавания?

- Подготовить базу обучающих фонограмм
- Использовать существующую акустическую модель для РАЗМЕТКИ фонограмм на отдельные состояния HMM
- Сопоставить каждому вектору признаков выходной one-hot вектор, в котором 1 соответствует состоянию, указанному разметкой на данном кадре
- Как правило, используется frame-stacking, чтобы учесть временной контекст
- Обучить ANN по этому набору входов и выходов
- Оценить априорные вероятности состояний $p(s_i)$.
  - Можно использовать частоты встречаемости состояний в разметке
  - Другой вариант: посчитать среднюю апостериорную вероятность данного состояния на всех наблюдениях обучающей базы.
- При необходимости переразметить и повторить

ГИБРИДНЫЕ И ТАНДЕМНЫЕ СИСТЕМЫ

Два исторически различных подхода:

- В тандемной системе нейронная сеть используется в качестве генератора новых высокоуровневых признаков, а классификатор – GMM
- Известные варианты тандемных систем:
  - TRAP-признаки (H.Hermansky, 2003)
  - LC-RC-признаки (P. Schwarz, 2008)
  - Bottleneck-признаки
- В гибридной системе ANN используются непосредственно в качестве классификаторов состояний.
  - Большинство современных систем – гибриды с разными типами ANN
  - Первый гибрид с NN-классификатором связанных состояний: CD-DNN-HMM (G.Dahl etal, 2011)

ГЛУБОКИЕ НЕЙРОННЫЕ СЕТИ И ИХ ОБУЧЕНИЕ

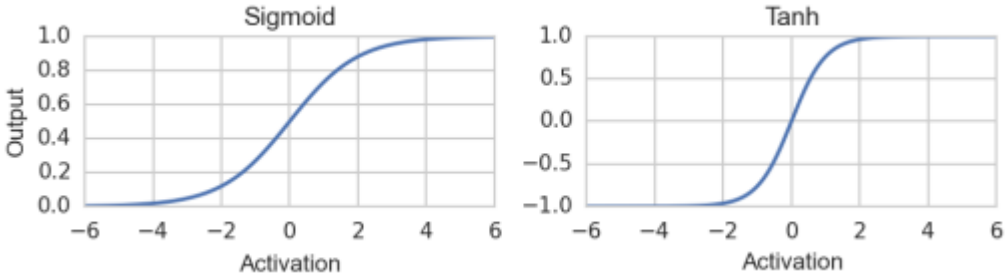
Глубокие нейронные сети (Deep Neural Networks, DNN)

- Глубокой называется нейронная сеть, у которой больше одного скрытого слоя
- Как правило под DNN подразумеваются полносвязные (fully-connected) сети прямого распространения (feed-forward).
- Размер входного слоя равен размеру входного вектора признаков (например, 13x11=143, если используются 11 кадров MFCC – 5 слева, текущий, 5 справа)
- Размер выходного слоя – число связанных состояний HMM
- Критерий обучения – средняя на кадр кросс-энтропия по всей базе: на текущем кадре определяется, как (y_i - i-й выход DNN, t_i - целевое значение)

$$CE = -\sum_{i=1}^{N} t_i \log y_i$$

Трудности обучения DNN

- Традиционный способ обучения DNN – стохастический градиентный спуск
- Традиционная реализация – метод обратного распространения ошибки (error back-propagation)
- Традиционные функции активации (до начала 2010-х) – логистическая сигмоида $\sigma(x) = \frac{1}{1+e^{-x}}$ или гиперболический тангенс $th(x) = \frac{e^{x}-e^{-x}}{e^{x}+e^{-x}}$:



- Производные этих функций активации быстро убывают при удалении от нуля

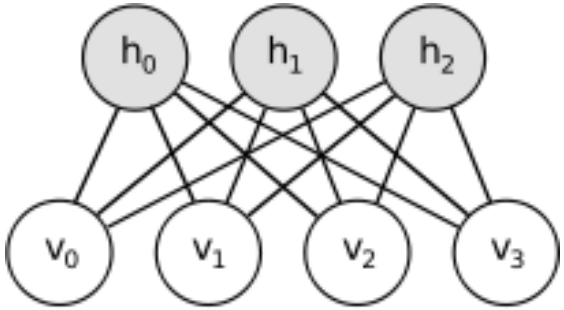
Трудности обучения DNN

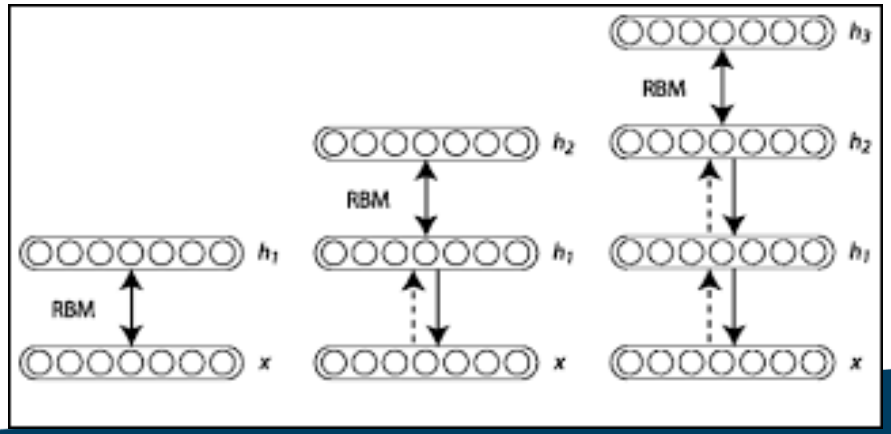
- Если в DNN много слоев с сигмоидальными активациями, до градиент ошибки быстро затухает при распространении от выходного слоя «вниз» по сети. Следовательно параметры изменяются очень слабо. Нижние слои НЕ УЧАТСЯ
- Это называется проблемой gradient vanishing.
- Возможные решения:
  - Предобучение (pretraining) сети: выбор хорошей стартовой точки для алгоритма стохастического градиентного спуска
  - Использование функций активации, производные которых не убывают

СПОСОБЫ ПРЕДОБУЧЕНИЯ DNN

Использование RBM (P.Smolensky, 1986, G.E.Hinton, 2006)

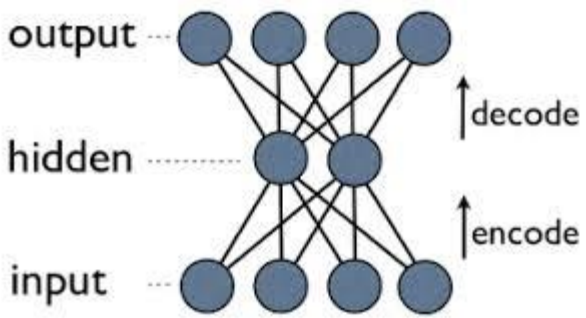
- RBM + Greedy layer-wise learning
- RBM (Restricted Boltzmann Machine) – это стохастическая НЕНАПРАВЛЕННАЯ нейронная сеть с одним скрытым слоем, которая учится воспроизводить распределение данных, на которых она обучалась.
- После обучения одной RBM вычисляются активации скрытого слоя для всех входных данных и на них, как на входных данных обучается следующая RBM





СПОСОБЫ ПРЕДОБУЧЕНИЯ DNN

Использование автоэнкодеров (Vincent etal. 2010)



- Автоэнкодер (autoencoder) – это нейронная сеть, которая учится на выходе восстанавливать свой вход. Обучается по критерию MSE.
- Выходы скрытого слоя содержат скрытое представление входных данных, позволяющее decoder-части восстанавливать входную информацию
- Шумоподавляющий (denoising) автоэнкодер (DAE) – это автоэнкодер, вход которого специально зашумляется перед подачей в сеть, а она стремится восстановить неискаженный (исходный) вход.
- Выходы скрытого слоя DAE можно использовать в качестве входов для обучения нового автоэнкодера.

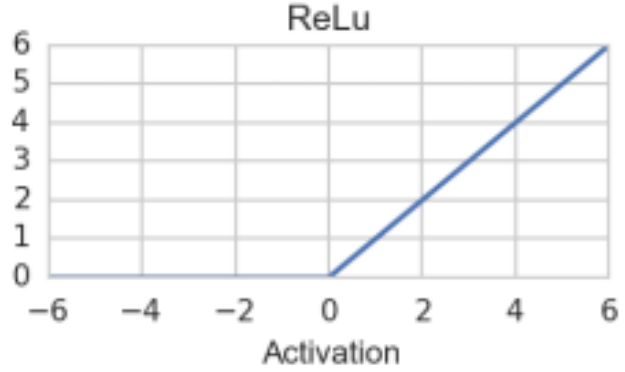
СПОСОБЫ ПРЕДОБУЧЕНИЯ DNN

Дискриминативное предобучение (F.Seide etal. 2011)

- Предобучение с помощью RBM и автоэнкодеров – без учителя (unsupervised)
- Но если есть целевые значения, то их можно использовать!
- Сначала обучаем сеть с одним скрытым слоем
- После этого «отрываем» классифицирующий слой (softmax), добавляем еще один слой и обучаем сеть с двумя скрытыми слоями (зафиксировав веса первого скрытого слоя)
- И т.д. В результате при обучении каждой следующей сети нижние слои уже ПРЕДОБУЧЕНЫ!

Кусочно-линейные функции активации

- Rectified linear unit (ReLU): $relu(x) = \max(x, 0)$



- Производная везде слева равна 0, а везде справа равна 1.
- Градиент при проходе через ReLU либо зануляется, либо не изменяется
- Используя нейроны с ReLU-активациями удается обучать глубокие нейронные БЕЗ ПРЕДОБУЧЕНИЯ!
- Есть много модификаций ReLU: leaky ReLU, noisy ReLU, ELU, maxout и т.д.
- С ReLU и ее аналогами надо внимательно выбирать скорость обучения!

ГЛУБОКИЕ НЕЙРОННЫЕ СЕТИ И ИХ ОБУЧЕНИЕ

CD-DNN-HMM-гибрид

- Гибридная модель, в качестве классификатора используется DNN
- Выходы нейронной сети соответствуют связанным состояниям трифонных HMM-моделей (т.е. требуется сначала обучить какие трифонные HMM со связыванием).
- Число слоев 5-7, размеры слоев 512-2048, число выходов ~5-20 тысяч.
- Вероятности перехода ИГНОРИРУЮТСЯ (считаются равными 0.5)
- Сравнение результатов на базе Switchboard Hub5 eval2000 (Seide etal. 2011):

Модель	WER на монофонах	WER на сенонах (9304)
CD-GMM-HMM (BMMI)	---	23.6%
CD-DNN-HMM (7x2048)	34.9%	17.1%

ГЛУБОКИЕ НЕЙРОННЫЕ СЕТИ И ИХ ОБУЧЕНИЕ

Sequence discriminative training (K.Vesely etal. 2013)

- Традиционное кросс-энтропийное обучение стремится уменьшить среднюю ошибку по кадрам (Frame Error Rate, FER)
- Но основной показатель для ASR – пословная ошибка (WER)
- Как обучить сеть, чтобы она меньше ошибалась в целой последовательности? Можно использовать дискриминативные критерии: MMI (минимизирует ошибку на целом предложении), MPE (минимизирует ошибку на фонемном уровне и т.д.
- Наиболее распространенный критерий – state-level Minimum Bayes Risk (sMBR) стремится минимизировать ошибку на уровне состояний HMM.
- Производные критерия по выходам сети легко вычисляются
- Типичное улучшение WER на 1.5-2%

ДРУГИЕ АРХИТЕКТУРЫ НЕЙРОННЫХ СЕТЕЙ

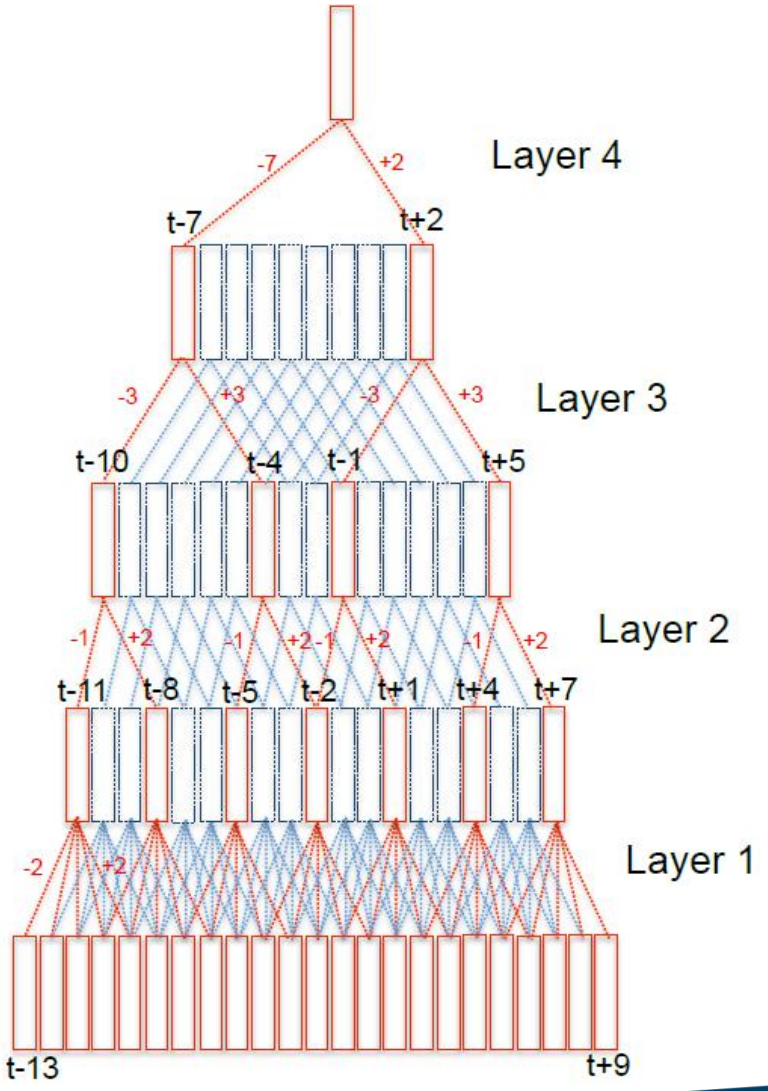
Какие еще нейронные сети применяют в ASR?

- Недостатки традиционных DNN:
  - Много параметров, необходимо много данных для обучения
  - Не в состоянии учитывать длинный временной контекст
- Альтернативные архитектуры
  - Нейронные сети с задержкой времени (Time Delay Neural Networks, TDNNs)
  - Сверточные нейронные сети (Convolutional Neural Networks (CNNs)
  - Рекуррентные нейронные сети (Recurrent Neural Networks, RNNs)
  - Различные комбинации вышеперечисленных

ДРУГИЕ АРХИТЕКТУРЫ НЕЙРОННЫХ СЕТЕЙ

Time Delay Neural Network (Peddinti etal., 2015)

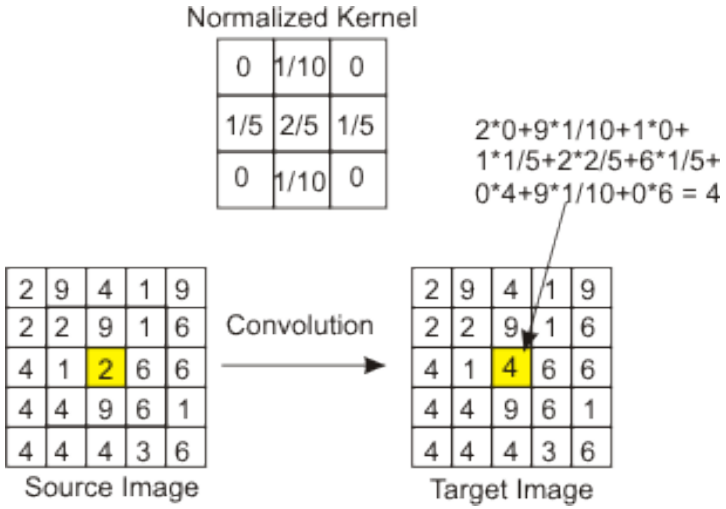
- Это обычная полносвязная сеть прямого распространения, но с хитрой топологией:
- Выходы каждого слоя накапливаются и на следующий слой подается набор из нескольких векторов (splicing)
- Благодаря этому можно обработать большой контекст не сильно увеличивая слои.

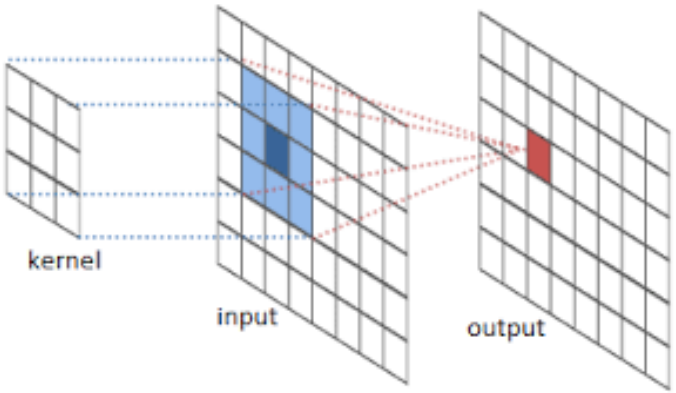


ДРУГИЕ АРХИТЕКТУРЫ НЕЙРОННЫХ СЕТЕЙ

Сверточные сети

- Широко используются для обработки изображений
- Содержат несколько сверточных слоев
  - Каждый сверточный слой – это набор «фильтров» (kernel – ядро свертки)
  - Каждый фильтр применяется к patch-ам входных данных, выходы фильтра образуют «карту» (map) признаков для подачи на следующий слой.

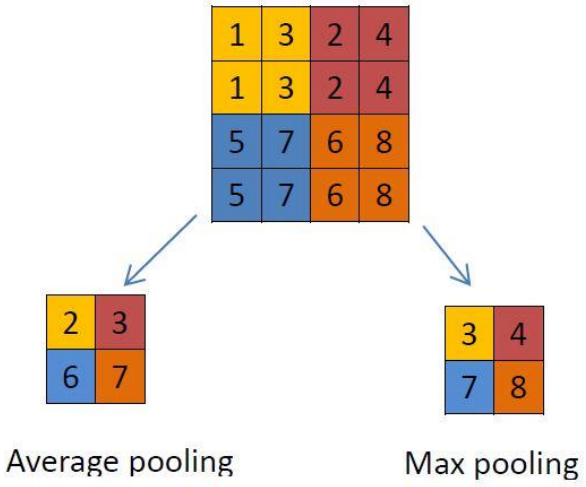




ДРУГИЕ АРХИТЕКТУРЫ НЕЙРОННЫХ СЕТЕЙ

Сверточные сети

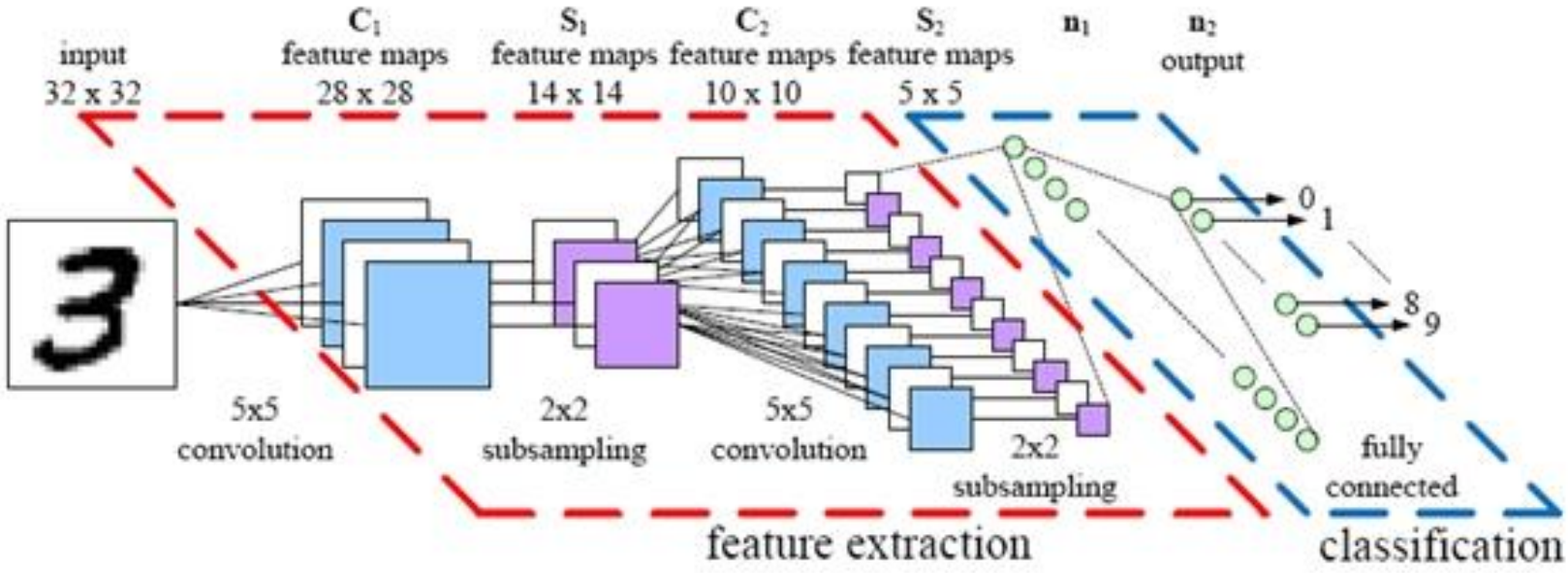
- Сверточные слои выделяют существенные локальные признаки
- После сверточного слоя часто следует слой пулинга (pooling)
  - Max-pooling – выход свертки обрабатывается подвижным окном, в каждом окне вычисляется максимальное значение.
  - Average pooling – то же самое, только вычисляется среднее



ДРУГИЕ АРХИТЕКТУРЫ НЕЙРОННЫХ СЕТЕЙ

Сверточные сети

- Типичная архитектура CNN (на примере изображений)



ДРУГИЕ АРХИТЕКТУРЫ НЕЙРОННЫХ СЕТЕЙ

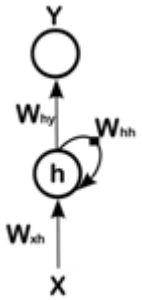
Достоинства CNN

- Сверточные слои содержат мало параметров (размер ядра обычно значительно меньше размера входных данных слоя и ядер не очень много)
- Сверточная часть сети выделяет высокоинформативные признаки, за счет чего достаточно несколько полносвязных слоев для хорошей классификации
- Выходы сверточных слоев можно использовать, как новые признаки для обучения более сложных классификаторов
- Сверточные сети обеспечивают определенную инвариантность к сдвигам и растяжению/сжатию (применительно к изображениям).
- В ASR сверточные сети применяются к спектру. Достигается определенная инвариантность к темпу речи и высоте голоса

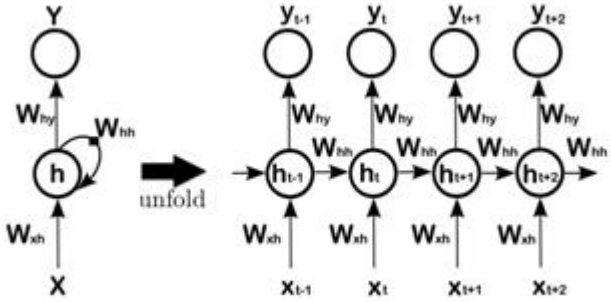
ДРУГИЕ АРХИТЕКТУРЫ НЕЙРОННЫХ СЕТЕЙ

Рекуррентные нейронные сети

- В отличие от сетей прямого распространения содержат обратные связи:
  - В нерекуррентной сети $\boldsymbol{h} = \boldsymbol{W_{xh}X}, \quad \boldsymbol{Y} = \boldsymbol{W_{hy}h}$
  - В рекуррентной сети $\boldsymbol{h} = \boldsymbol{W_{xh}X} + \boldsymbol{W_{hh}h}, \quad \boldsymbol{Y} = \boldsymbol{W_{hy}h}$



- Таким образом, скрытый слой содержит «состояние» сети которое динамически меняется в зависимости от входов!
- Для обучения RNN используется «развертывание» во времени



- Алгоритм обучения: backpropagation through time (BPTT)

ДРУГИЕ АРХИТЕКТУРЫ НЕЙРОННЫХ СЕТЕЙ

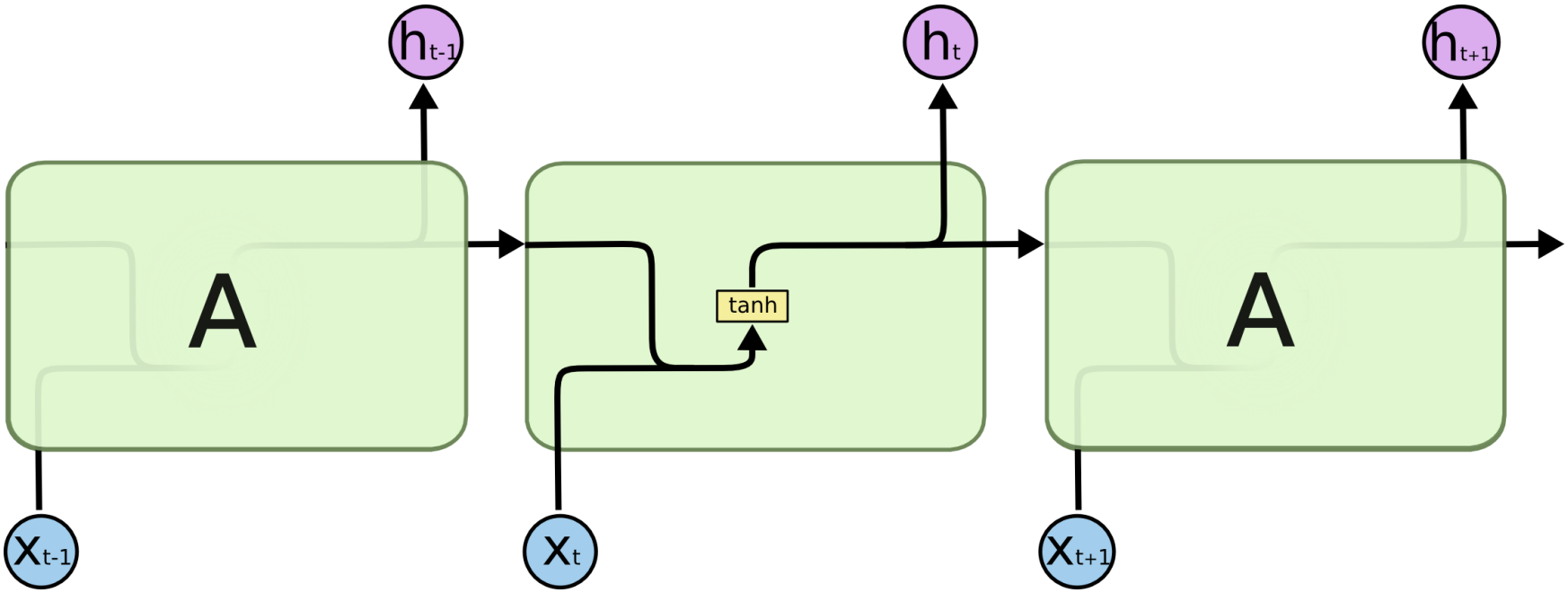
Рекуррентные нейронные сети: трудности обучения

- Временные затраты на обучение прямо пропорциональны длине развертывания сети
- За счет многократного умножения градиента на производные активаций скрытых слоев могут возникнуть две проблемы:
  - Gradient vanishing ведет к тому, что обучение почти останавливается, не дойдя до оптимума
  - Gradient explosion ведет к тому, что обучение расходится
- Существуют несколько различных подходов к преодолению этих проблем
- Наиболее распространенный вариант – использовать специальную архитектуру сети с памятью, которая называется LSTM (Long Short-Term Memory).

ДРУГИЕ АРХИТЕКТУРЫ НЕЙРОННЫХ СЕТЕЙ

LSTM (Hochreiter & Schmidhuber, 1997)

- Простая RNN (в развернутом состоянии):

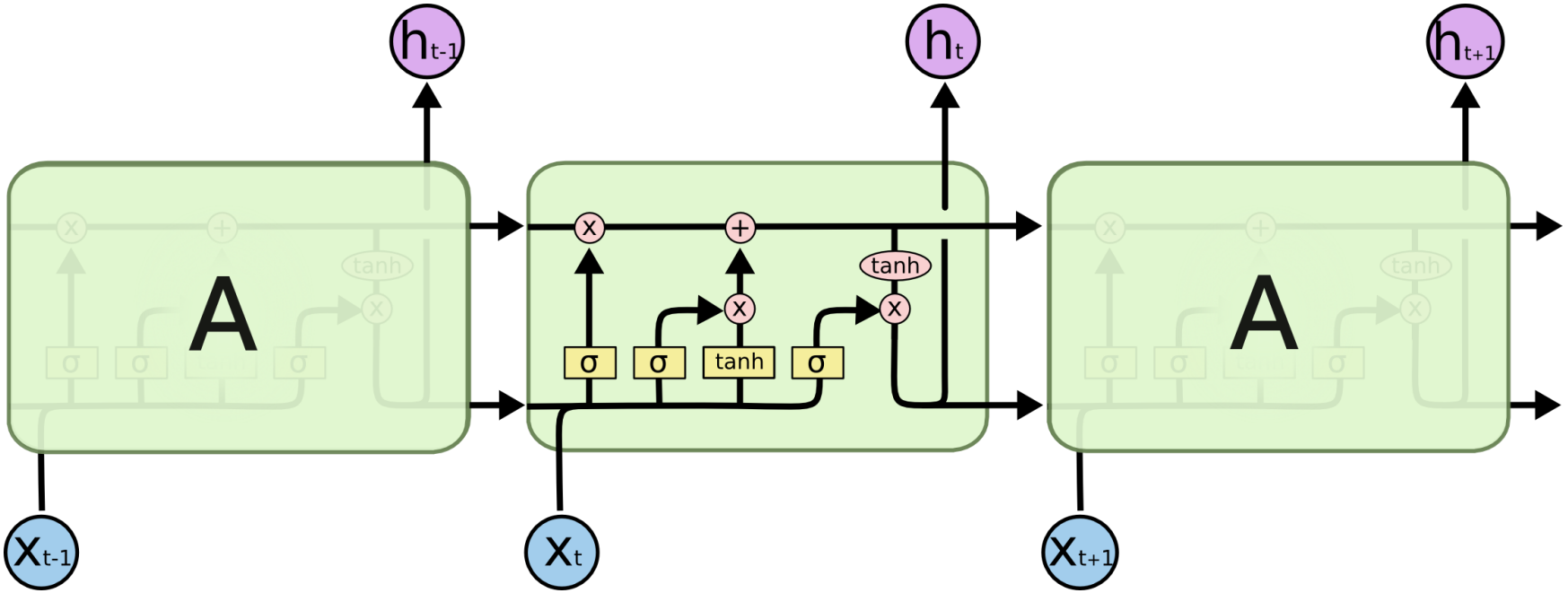


*Картинки позаимствованы из блога http://colah.github.io/posts/2015-08-Understanding-LSTMs/

ДРУГИЕ АРХИТЕКТУРЫ НЕЙРОННЫХ СЕТЕЙ

LSTM (Hochreiter & Schmidhuber, 1997)

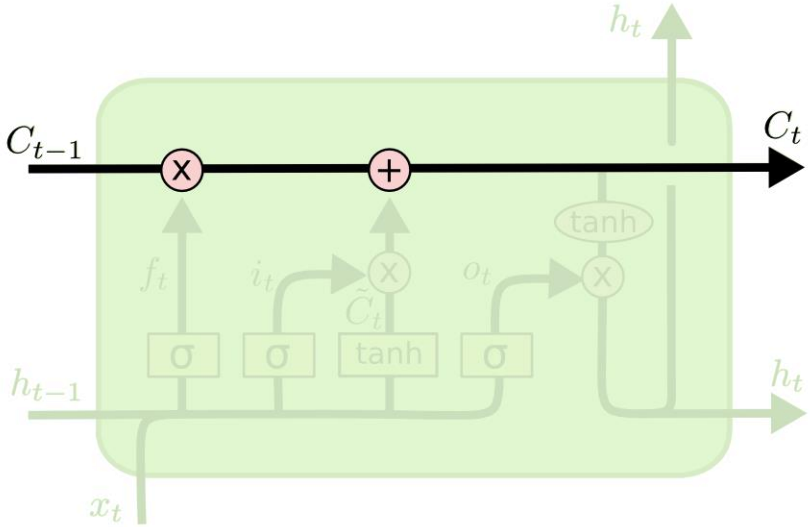
- LSTM (в развернутом состоянии):



ДРУГИЕ АРХИТЕКТУРЫ НЕЙРОННЫХ СЕТЕЙ

LSTM (Hochreiter & Schmidhuber, 1997)

- Ключевой элемент «памяти» LSTM - cell

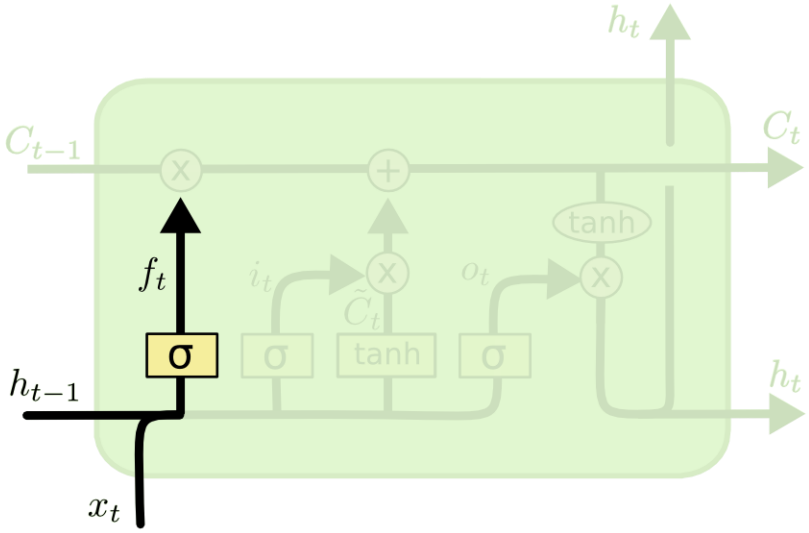


- Потоки информации через LSTM регулируются тремя «гейтами» (gate), которые представляют собой обычные сигмоидальные слои, выходы которых покомпонентно умножаются на векторы, проходящие через gate

ДРУГИЕ АРХИТЕКТУРЫ НЕЙРОННЫХ СЕТЕЙ

LSTM (Hochreiter & Schmidhuber, 1997)

- Forget gate принимает на вход очередной входной вектор и вектор состояния, запомненный на предыдущем шаге, и определяет, какую долю cell-информации следует пропустить дальше

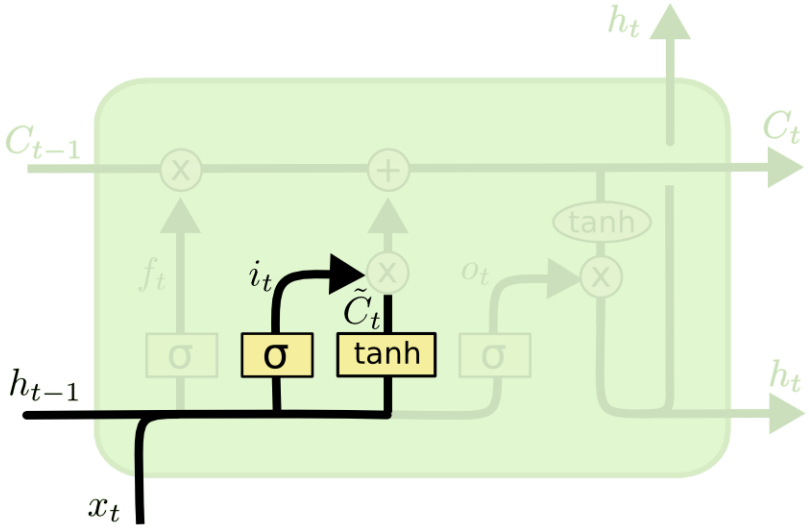


$$f_t = \sigma\left(W_f \cdot [h_{t-1}, x_t] \; + \; b_f\right)$$

ДРУГИЕ АРХИТЕКТУРЫ НЕЙРОННЫХ СЕТЕЙ

LSTM (Hochreiter & Schmidhuber, 1997)

- Input gate на основании той же входной информации определяет, какую долю новых данных следует запомнить в cell на данном кадре



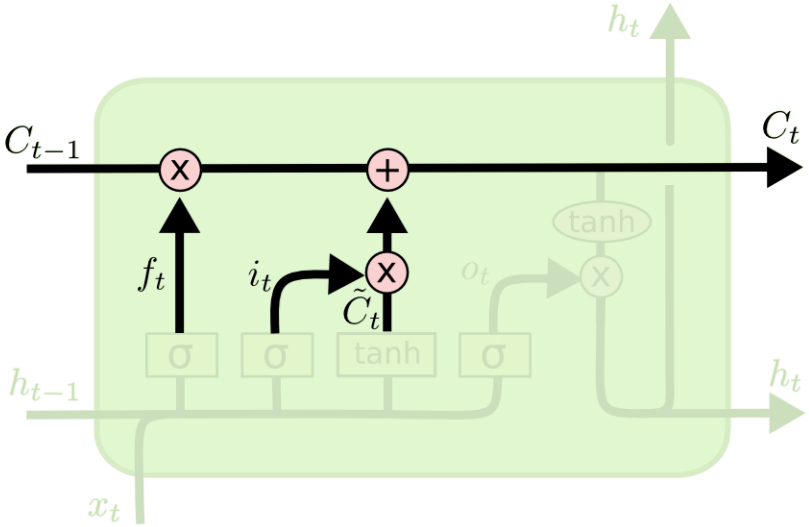
$$i_t = \sigma\left(W_i \cdot [h_{t-1}, x_t] \; + \; b_i\right)$$

$$\tilde{C}_t = \tanh(W_C \cdot [h_{t-1}, x_t] \; + \; b_C)$$

ДРУГИЕ АРХИТЕКТУРЫ НЕЙРОННЫХ СЕТЕЙ

LSTM (Hochreiter & Schmidhuber, 1997)

- Информация с выходов forget gate и input gate объединяется и формирует новое значение cell для передачи на следующий кадр:

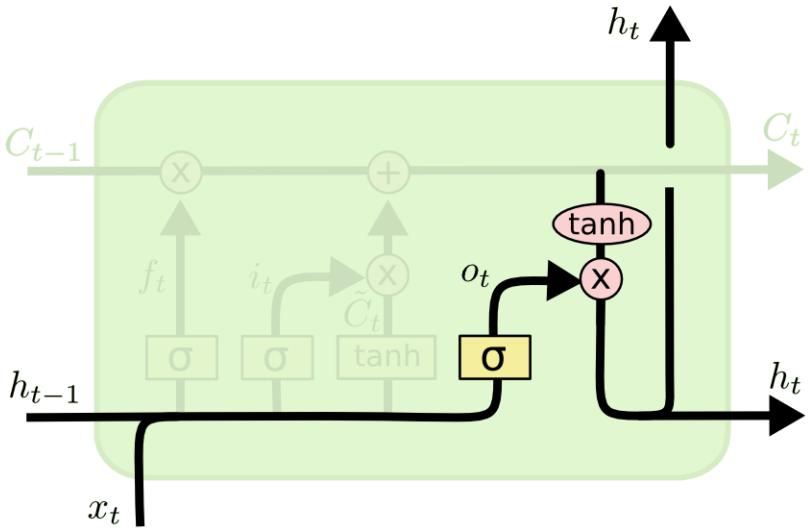


$$C_t = f_t * C_{t-1} + i_t * \tilde{C}_t$$

ДРУГИЕ АРХИТЕКТУРЫ НЕЙРОННЫХ СЕТЕЙ

LSTM (Hochreiter & Schmidhuber, 1997)

- Наконец, output gate определяет, какую долю вновь вычисленной информации из cell следует передать на следующий кадр в качестве вектора состояния:



$$o_t = \sigma\left(W_o\left[h_{t-1}, x_t\right] + b_o\right)$$

$$h_t = o_t * \tanh\left(C_t\right)$$

- Этот же вектор состояния является выходом слоя на данном кадре

ДРУГИЕ АРХИТЕКТУРЫ НЕЙРОННЫХ СЕТЕЙ

LSTM (Hochreiter & Schmidhuber, 1997)

- LSTM демонстрируют исключительно высокую производительность при обработке последовательных данных (текст, речь, видео)
- На основе LSTM создаются системы
  - Распознавания рукописного текста
  - Машинного перевода
  - Распознавания речи
  - Распознавания действий человека на видео
  - Языковые модели и т.д.
- Распространенные варианты/аналоги LSTM
  - GRU (Gated Recurrent Unit) - более простая структура, обеспечивающая похожие свойства
  - BLSTM (Bidirectional LSTM) – последовательность обрабатывается двумя LSTM-слоями в двух ПРОТИВОПОЛОЖНЫХ направлениях. После этого выходы обоих слоев объединяются.

END-TO-END СИСТЕМЫ РАСПОЗНАВАНИЯ

Недостатки традиционных CD-?NN-HMM систем

- Слишком многоуровневый процесс обучения:
  - Обучить GMM-HMM систему
  - Построить дерево решений и получить набор связанных состояний (сенонов)
  - Провести разметку обучающей базы на сеноны
  - Обучить нейронную сеть - классификатор сенонов
  - После этого еще декодировать по стейтовому графу
- Хочется чего-то более простого: подать в систему обучения звук (акустические признаки) и тексты (может быть, еще словарь) и получить обученную модель
- На этапе распознавания подать в систему звук (акустические признаки) и сразу получить на выходе последовательность слов (или хотя бы фонем).
- Такие системы называются end-to-end (E2E)

END-TO-END СИСТЕМЫ РАСПОЗНАВАНИЯ

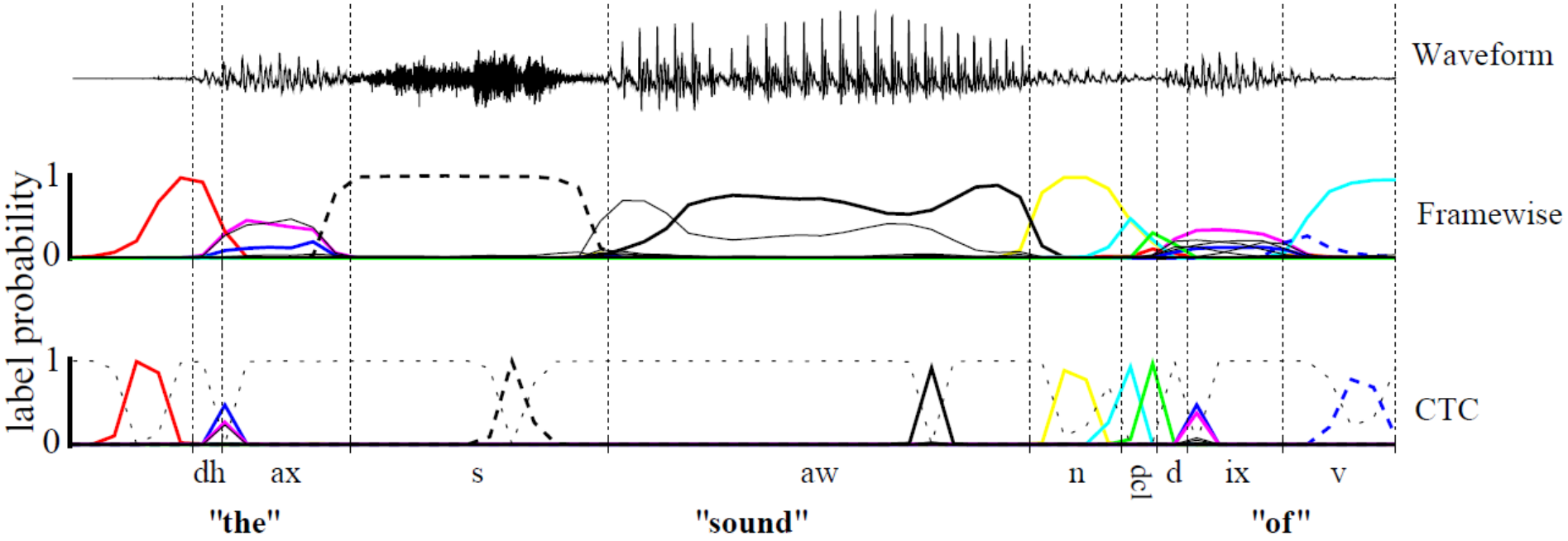
End-to-end система на основе CTC (A.Graves etal. 2006)

- В качестве моделей для E2E в большинстве случаев используется BLSTM
- Выходами BLSTM являются вероятности символов «алфавита» (фонем) и дополнительного пустого символа «blank», который означает ПРОДОЛЖЕНИЕ текущей фонемы.
- Для построения E2E системы используют специальную целевую функцию, CTC (Connectionist Temporal Classification) loss, вычисляемую с помощью Forward-Backward алгоритма, аналогичного используемому в HMM.
- Градиенты CTC loss по выходам нейронной сети можно вычислить, следовательно сеть можно обучать с помощью BPTT.

END-TO-END СИСТЕМЫ РАСПОЗНАВАНИЯ

Свойства CTC-систем:

- Иллюстрация из статьи Graves etal. 2006:



- Качество фонемных CTC-систем пока недотягивает до CD-DNN-HMM
- В 2016-м Google обучил словную CTC-систему на 125000 часов речи!

СПАСИБО ЗА ВНИМАНИЕ!

О КОМПАНИИ

ООО «Центр речевых технологий» (ЦРТ) — российская компания с 25-летней историей. За это время компания накопила богатейший научный потенциал и стала абсолютным лидером российского и значимым игроком международного рынка речевых технологий и мультимодальной биометрии.

Сегодня ЦРТ является ведущим мировым разработчиком инновационных систем в сфере высококачественной записи, обработки и анализа аудио-видео информации, синтеза и распознавания речи. Создаваемые в ЦРТ биометрические решения обеспечивают высокую точность распознавания личности по голосу и изображению лица в реальном времени. Эти решения находят успешное применение в государственном и коммерческом секторе, от небольших экспертных лабораторий до сложных систем безопасности национального масштаба.

Качество работы компании подтверждается сертификатами соответствия системы менеджмента качества требованиям международного стандарта ISO-9001:2008, а также государственного военного стандарта ГОСТ ISO 9001-2011 и ГОСТ РВ 0015-002-2012.

КОНТАКТНАЯ ИНФОРМАЦИЯ

Санкт-Петербург
Адрес: Санкт-Петербург, ул. Красуцкого, 4
Телефон: (+7 812) 325-88-48
Факс: (+7 812) 327-92-97
Отдел продаж: (+7 812) 325-88-48 доб.1
Эл. почта: stc-spb@speechpro.com
Почтовый адрес: 196084 Санкт-Петербург а/я 124 «Центр речевых технологий»

Москва
Адрес: Москва, ул. Марксистская, д.3, стр.2, Бизнес-центр "Таганский"
Телефон: (+7 495) 669-74-40
Факс: (+7 495) 669-74-44
Эл. почта: stc-msk@speechpro.com