Центр
Речевых
Технологий

Применение технологии CUDA для задач голосовой биометрии на примере построения универсальной фоновой модели диктора

В.В. Габдуллин, А.И. Капустин,
А.И. Королев

Докладчик: В.В. Габдуллин

Цели и задачи

Цели:

- уменьшение времени обучения голосовой биометрической системы;
- повышение надежности биометрической системы.

Задачи:

- разработка параллельного алгоритма обучения универсальной фоновой модели на CPU и GPU;
- сравнение быстродействия и точности алгоритмов на CPU и GPU;
- анализ эффективности реализации алгоритмов;
- обучение универсальной фоновой модели на большом объеме данных.

Задачи голосовой биометрии

Задачи, связанные с определением пользователя по голосу, можно разделить на:

- **идентификацию** – задача состоит в классификации речевого сигнала, при этом каждый класс соответствует одному человеку, зарегистрированному в системе;
- **верификацию** – задача представляет собой бинарную классификацию.

Более формально, задача верификации представляет собой проверку двух гипотез:

H0: фразу Y произнес диктор S

H1: фразу Y произнес НЕ диктор S.

При этом, процедура принятия решения выглядит следующим образом:

$$\frac{p(Y \mid H_0)}{p(Y \mid H_1)} \begin{cases} \geq \theta, & H_0 \\ < \theta, & \overline{H}_0, \end{cases} \qquad (1)$$

где p(Y|H) – функция плотности вероятности для гипотезы H, оцененная на речевом сегменте Y, а θ – порог принятия решения.

Модель гауссовых смесей

(GMM – gaussian mixture model)

Математически гипотеза **H** может быть определена моделью ***λ***, которая характеризует диктора **S** в пространстве признаков.

В последнее время для верификации личности диктора по голосу применяется модель гауссовых смесей. Для D-мерного вектора признаков x, функция плотности распределения описывается следующей функцией:

$$p(x \mid \lambda) = \sum_{i=1}^{M} \omega_i \, p_i(x) \qquad (2)$$

где ***M*** – количество компонент, ω_i – вес i-ой компоненты, а $p_i(x)$ – плонтость распределения каждой компоненты, которая представляет собой **D**-мерный гауссиан:

$$p_i(x) = \frac{1}{(2\pi)^{D/2}\left|\sigma_i\right|^{1/2}} \exp\left\{-1/2(x-\mu_i)'\sigma_i^{-1}(x-\mu_i)\right\} \qquad (3)$$

где μ_i – вектор математического ожидания, а σ_i – ковариационная матрица.

Плотность распределения смеси Гауссиан полностью описывается параметрами компонент и значениями весов и представляются как $\lambda=\{\omega_i, \sigma_i, \mu_i\}$

Универсальная фоновая модель
(UBM – universal background model)

• Для гипотезы **H1** строится универсальная фоновая модель, цель которой характеризовать всех возможных говорящих во всех возможных контекстах.

• Данная модель обучается на очень большом количестве речевых данных, сбалансированных по гендерному типу, а также по оборудованию и стандартным условиям.

• Для существующих систем идентификации используются базы речевых данных в несколько сотен часов, при этом обучение UBM модели может длиться не одну неделю на современном CPU.

• Для определения параметров модели λ существуют несколько методов, наиболее распространенным из которых является метод максимального правдоподобия.

Обучение GMM-UBM

метод максимального правдоподобия (EM – expectation-maximization)

Задача метода состоит в нахождении по заданным обучающим данным таких параметров модели, при которых функция правдоподобия модели достигает максимума.

Для последовательности из T обучающих векторов $X=\{x_1,x_2 \ldots x_i\}$ функция правдоподобия имеет вид:

$$p(X \mid \lambda) = \sum_{t=1}^{T} p(x_t \mid \lambda) \qquad (4)$$

На первом шаге алгоритма (expectation) вычисляется ожидаемое значение функции правдоподобия:

$$\Pr(i \mid x_t) = \frac{\omega_i p_i(x_t)}{\sum_{j=1}^{M} \omega_j p_j(x_t)} \qquad (5)$$

Обучение GMM-UBM (EM алгоритм)

На втором шаге (maximization) вычисляется оценка максимального правдоподобия для каждой компоненты модели:

$$n_i = \sum_{t=1}^{T} \Pr(i \mid x_t) \tag{6}$$

$$E_i(x) = \frac{1}{n_i} \sum_{t=1}^{T} \Pr(i \mid x_t) x_t \tag{7}$$

$$E_i(x^2) = \frac{1}{n_i} \sum_{t=1}^{T} \Pr(i \mid x_t) x_t^2 \tag{8}$$

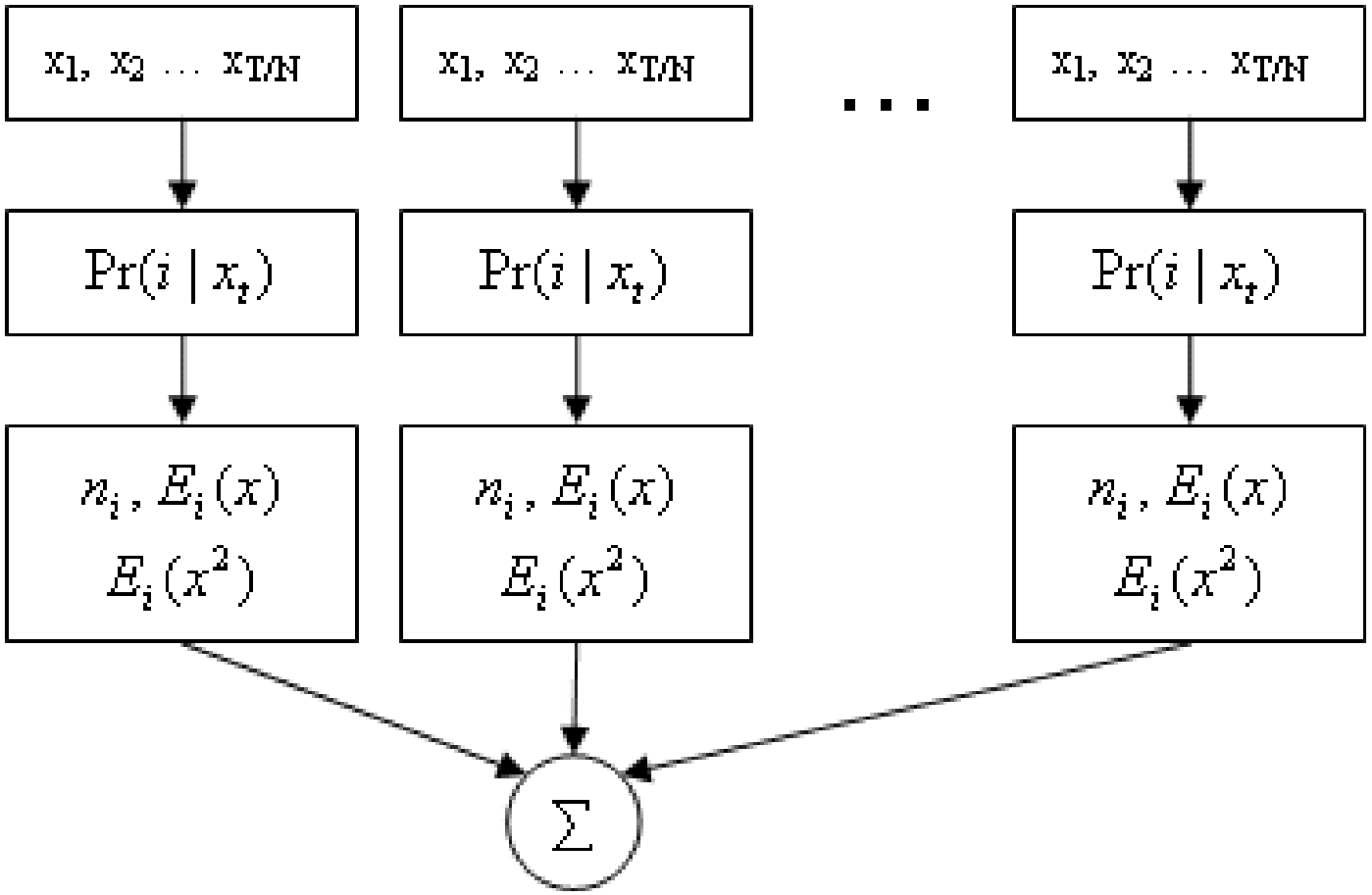
Затем вычисляются новые параметры модели, которые используются на первом шаге следующей итерации алгоритма:

$$\hat{\omega}_i = n_i / T \tag{9}$$

$$\hat{\mu}_i = E_i(x) + \mu_i \tag{10}$$

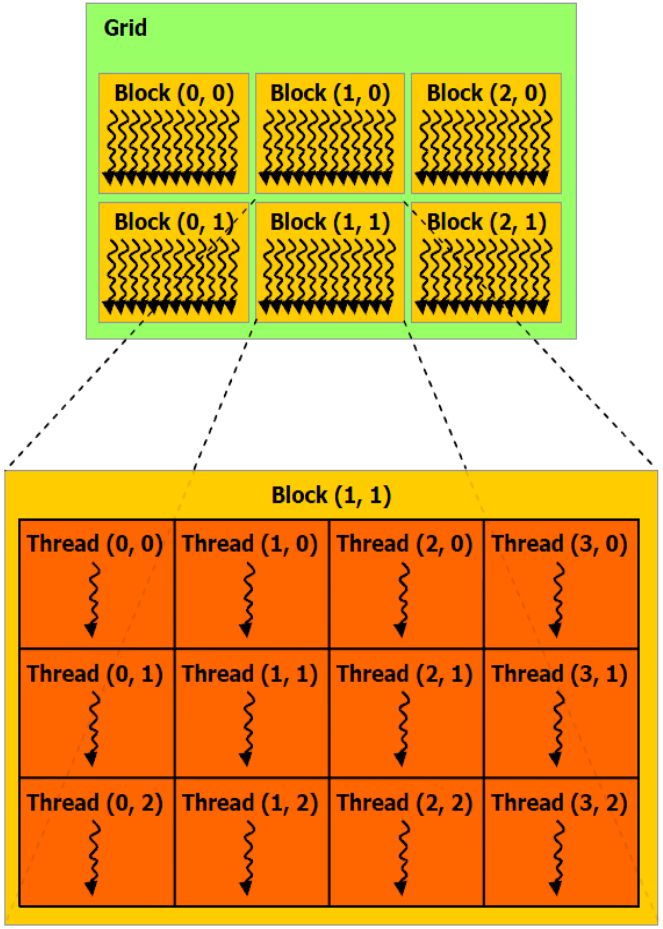
$$\hat{\sigma}_i = E_i(x^2) + (\sigma_i^2 + \mu_i^2) - \hat{\mu}_i^2 \tag{11}$$

Параллельное обучение UBM



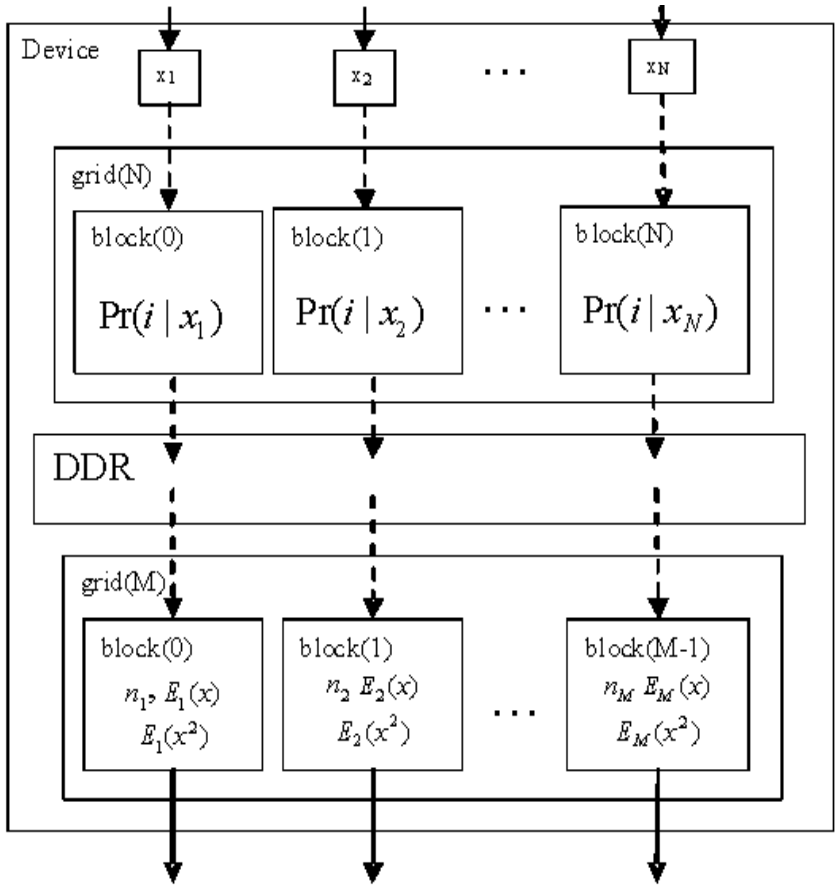
Особенности архитектуры CUDA

- GPU выступает в роли массивно-параллельного сопроцессора к CPU;
- нити на GPU обладают крайне небольшой стоимостью создания, управления и уничтожения;
- основным местом размещения и хранения большого объема данных, для обработки ядрами, является глобальная память, размещаемая в DRAM GPU;
- все запущенные на выполнение нити организованы в многоуровневую иерархию;
- каждый блок получает в свое распоряжение 48 кб быстрой разделяемой памяти, которую все нити блока могут совместно использовать.



Параллельное обучение UBM на GPU (CUDA)

- для вычисления функции правдоподобия используется быстрая память;
- первый и второй шаг алгоритма EM выполняется раздельно для максимального использования потоков (атомарные функции аккумуляции пока недоступны для float и double);
- промежуточные результаты сохраняются в DDR памяти GPU;
- минимальное копирование данных по шине PCI-Express (только обучающие векторы и результат итерации).



Конфигурация испытательных стендов

Блок	**X2_GTX285**	**X16_GTX480**
Процессор	Intel Core2 Duo E6550	Intel Xeon E5630 Intel Xeon E5630
Число ядер	2	16 (8 + hyper threading)
Объем RAM	2 ГБ	48 ГБ
Видеокарта	GeForce GTX 285	GeForce GTX 480
Число ядер CUDA	240	480
Разделяемая память GPU	16 кБ	48 кБ

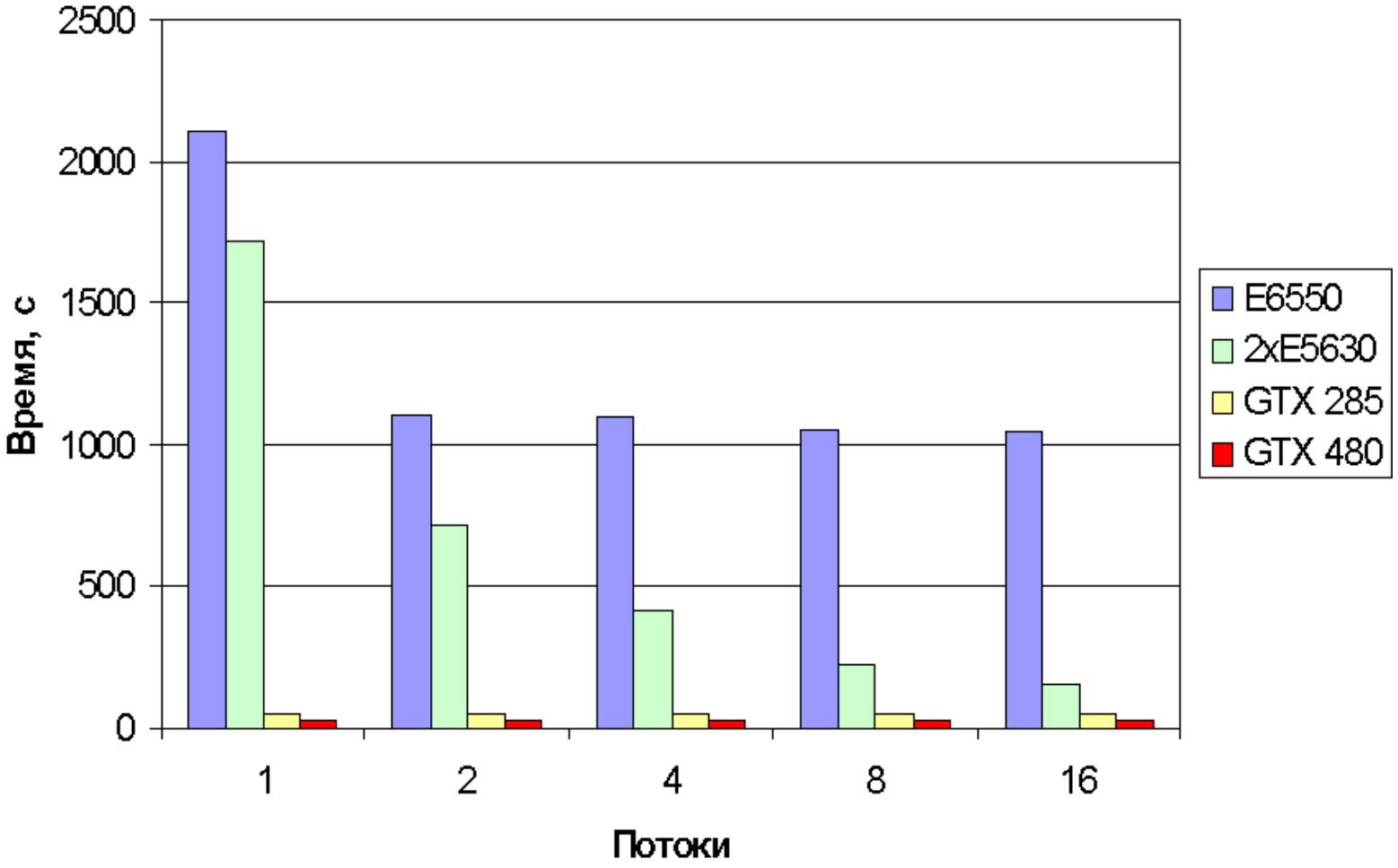
Результаты испытаний

(100 файлов, ~10000 обучающих векторов в каждом, 10 итераций обучения)

	UBM 512 Xeon E5630		UBM 512 Core 2 E6550		UBM 1024 Xeon E5630	
Потоки	Время, с	Расчеты, с	Время, с	Расчеты, с	Время, с	Расчеты, с
1	1727	1716	2134	2111	3337	3328
2	725	718	1116	1105	1424	1416
4	423	415	1115	1094	842	830
8	235	225	1087	1054	458	441
16	165,7	155	1093	1043	352	333
	UBM 512 GTX 480		UBM 512 GTX 285		UBM 1024 GTX 480	
	45,5	23,3	72,7	53,1	723	57,5

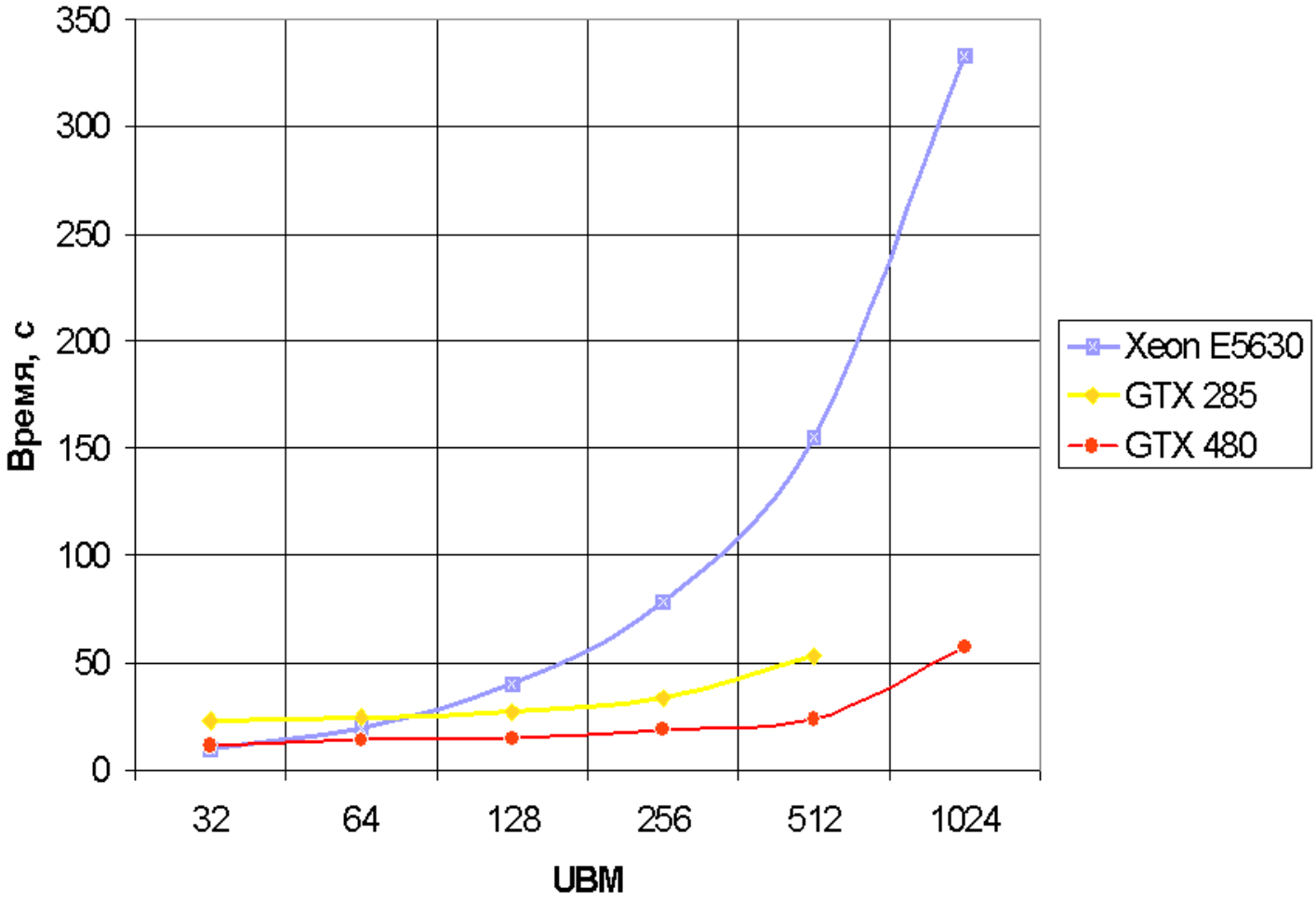
Зависимость времени вычисления от числа потоков

(100 файлов, ~10000 обучающих векторов в каждом, 10 итераций обучения)



Зависимость времени вычисления от размера UBM

(100 файлов, ~10000 обучающих векторов в каждом, 10 итераций обучения)



Эффективность вычислений

(100 файлов, ~10000 обучающих векторов в каждом, 10 итераций обучения
1024 гауссоиды)

Объем вычислений: 8 TFLOP
Объем данных: 9.6 ТБ
Стенд: X16_GTX480

Устройство	Ресурс	Теоретическое значение	Теоретическое время вычислений	Реальное время вычислений	% использования ресурса
CPU	Вычисления	320 GFLOP	25 с	333 с	7,5% (30%)
	Память	25,6 ГБ/с	376 с		100% (cache)
GPU	Вычисления	1,5 TFLOP	5,3 с	57,5 с	10,9%
	Память	177 ГБ/с	54,2 с		94%

Точность вычислений

(146 часов, ~15,7 млн. обучающих векторов, 20 итераций обучения, 512 гауссоид)

В качестве критерия оценки использовалась величина равновероятной ошибки EER:

$$EER = FR(\theta) = FA(\theta) \quad (12)$$

База	EER(CPU, float), %	EER(GPU, float),%	EER(GPU, double), %
Микрофон мужчины	4,4	4,2	4,2
Микрофон женщины	3,4	4,4	4,2
Телефон мужчины	12,6	11,6	11,6
Телефон женщины	10,5	10,4	10,6

Результаты

- Реализованы параллельные алгоритмы обучения UBM на CPU и GPU с поддержкой технологии CUDA;
- Эффективность использования памяти GPU – 94%;
- Время обучения UBM уменьшено:
  - 512 гауссоиды: в 3,6 (**6,65**) раза;
  - 1024 гауссоиды: в 4,9 (**5,8**) раза;
- Погрешность вычислений на результат влияет не существенно;
- Увеличение объема данных, участвующего в обучении системы, позволило существенно улучшить надежность (EER **2%**)

Перспективы

- Использование ресурсов нескольких GPU, скорость вычислений будет наращиваться линейно
- Оптимизация чтения обучающих векторов с диска
- Перенос части данных в константную память GPU
- Повышение скорости сравнения моделей в больших биометрических системах

Спасибо за внимание!