Добавляем большую языковую модель (LLM) в приложение на C++

с помощью llama.cpp на реальном примере

Кирилл Колодяжный

Обо мне

- Инженер-программист отдела оптимизации производительности СХД в YADRO
- Автор книги «Hands-On Machine Learning with C++»
- Ранее: геймдев, CV, ML, алгоритмы 3D-реконструкции, веб-браузеры

Одно и тоже разными словами

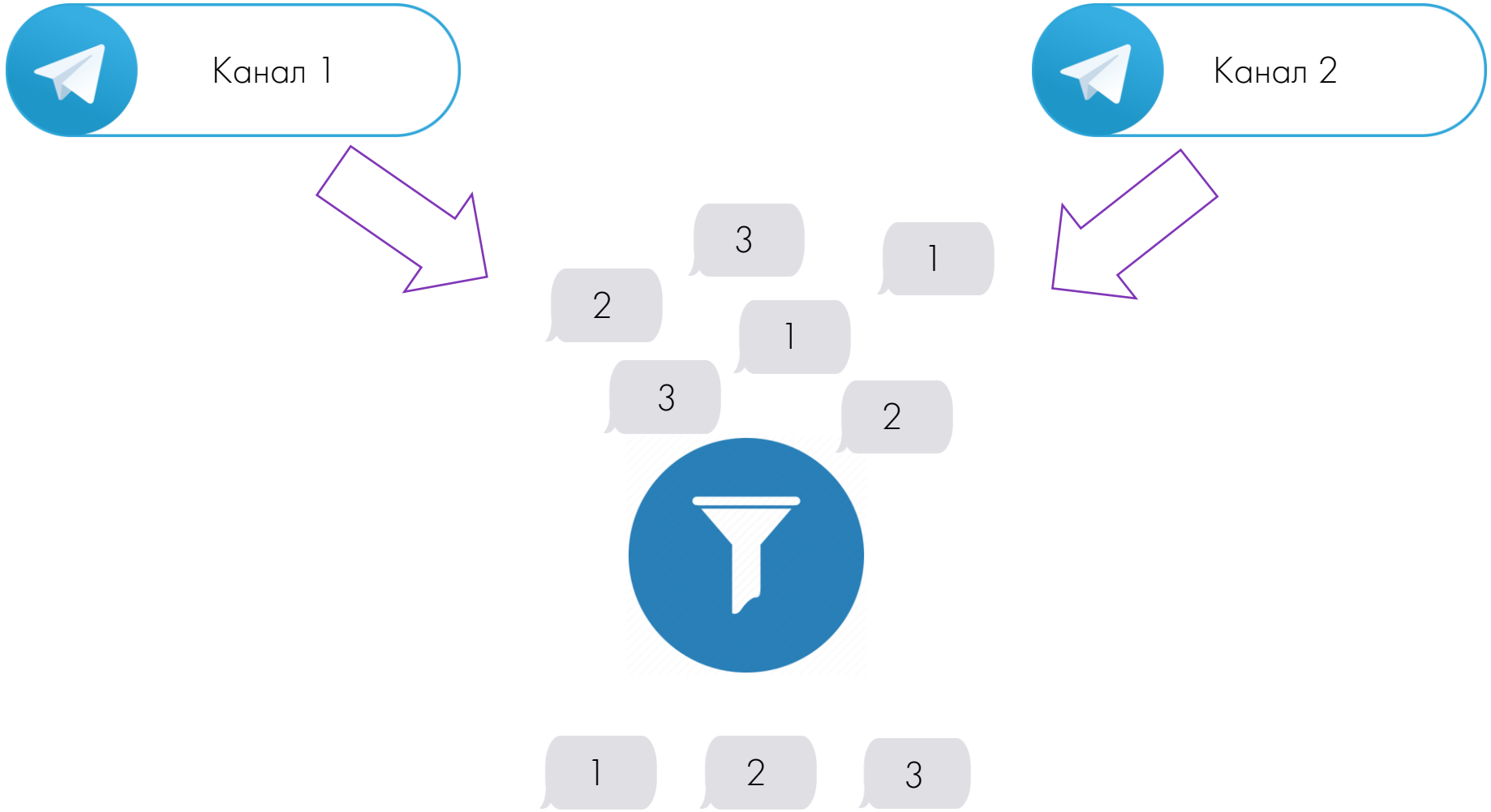
ё-Пром:

Пензенское предприятие «Электроприбор» на 20% увеличило эффективность использования станков с ЧПУ на фрезерном и токарном участках за счет внедрения системы планирования ресурсов (ERP) «Платон» собственной разработки, сообщил материнский холдинг «Росэлектроника».

Время Вперёд!:

В Пензе на Производственном объединении «Электроприбор» создали и внедрили собственную ERP-систему «Платон», что привело к увеличению эффективности работы станков с числовым программным управлением (ЧПУ) на фрезерном и токарном участках.

Решение – свой клиент Telegram



Этапы создания приложения

1. Выбор инструментов
2. Решение задачи обобщения текстов
3. Решение проблемы интеграции инструментов
4. Решение задачи сравнения текстов
5. Объединение всех элементов

Выбор инструментов

Классические подходы

- Частота встречаемости
- Индекс поиска
- TF-IDF
- Мешок слов
- ...

vs

Современные методы

LLM

Требования к работе LLM

1. Запускать локально без доступа к серверам
2. Использовать готовые модели без дополнительной тренировки

Локальная LLM

- llama.cpp – запуск квантованных моделей LLaMA
- Чистый C/C++ без зависимостей
- Оптимизация для x86, Apple, ARM
- Mixed F16 / F32 precision
- 2, 3, 4, 5, 6, 8 битовые квантования
- Поддержка CUDA, Metal и OpenCL

Понимают ли LLM сообщения?

Текст:

Our HDD benchmarks hierarchy shows all of the high-capacity hard drives that we've tested over the years, ranked in order of overall sequential throughput. It's a companion to our guide of the best hard drives, as well as the best SSDs and the SSD benchmarks hierarchy — we strongly recommend any modern PC use an SSD for the OS and boot drive, while HDDs are best as secondary storage.

Обобщение:

The HDD benchmarks hierarchy shows the various hard drives that have been tested by the site over the years, and are ranked according to the sequential throughput. This hierarchy is a companion to the best hard drives guide. The best hard drives are the SSDs.

Промпт для обобщения

```
### Задача:

Напиши краткое содержание следующего текста.

### Текст:

Текст сообщения

### Ответ:
```

Проблема с русским языком

Текст:

Компания Lenovo выпустила очередное поколение ноутбука ThinkPad P1 (Gen 7), который является достаточно необычным с технической точки зрения и в целом уникальным на данный момент. Суть в том, что впервые на рынке он сочетает новейшие процессоры Intel Core Ultra и пока ещё очень редко встречающуюся оперативную память LPCAMM2.

Обобщение:

Научно-технический ноутбук Lenovo ThinkPad P1 (Gen 7) сочетает в себе самые новые процессоры Intel Core Ultra и оперативную память LPCAMM2.

Saiga – решение проблемы работы с русским языком

Saiga – пример обработки

Текст:

Компания Lenovo выпустила очередное поколение ноутбука ThinkPad P1 (Gen 7), который является достаточно необычным с технической точки зрения и в целом уникальным на данный момент. Суть в том, что впервые на рынке он сочетает новейшие процессоры Intel Core Ultra и пока ещё очень редко встречающуюся оперативную память LPCAMM2.

Обобщение:

Компания Lenovo выпустила новый ноутбук ThinkPad P1 (Gen 7), который уникален благодаря сочетанию новейших процессоров Intel Core Ultra и редкой оперативной памяти LPCAMM2.

Интеграция llama.cpp в приложение

Основные компоненты:

- GGML – библиотека тензорной алгебры и блоков для нейросетей
- llama – библиотека с архитектурой LlaMa
- common – библиотека облегчает работу с библиотекой llama
- Приложение

Библиотека GGML

```
// f(x) = a*x^2 + b
ggml_init_params params = {
    .mem_size   = 16*1024*1024,
    …
};
ggml_context* ctx = ggml_init(params);
ggml_tensor* x  = ggml_new_tensor_1d(ctx, GGML_TYPE_F32, 1);
ggml_tensor* a  = ggml_new_tensor_1d(ctx, GGML_TYPE_F32, 1);
ggml_tensor* b  = ggml_new_tensor_1d(ctx, GGML_TYPE_F32, 1);
ggml_tensor* x2 = ggml_mul(ctx, x, x);
ggml_tensor* f  = ggml_add(ctx, ggml_mul(ctx, a, x2), b);
```

Библиотека llama

```
LLAMA_API struct llama_model * llama_load_model_from_file(const char * path_model,
                                                          struct llama_model_params params);

LLAMA_API int32_t llama_tokenize(const struct llama_model * model,
                                 const char * text,
                                 int32_t   text_len,
                                 llama_token * tokens,
                                 int32_t   n_max_tokens,
                                 bool   add_bos,
                                 bool   special);
```

Библиотека common

```
std::tuple<llama_model *, llama_context *> llama_init_from_gpt_params(gpt_params & params);

std::vector<llama_token> llama_tokenize(const struct llama_context * ctx,
                                        const std::string & text,
                                        bool   add_bos,
                                        bool   special = false);

llama_token llama_sampling_sample(struct llama_sampling_context * ctx_sampling,
                                  struct llama_context * ctx_main,
                                  struct llama_context * ctx_cfg,
                                  int idx = 0);
```

Приложение llama.cpp

```
root@a6c09d00d7a8:/llamacpp/llama.cpp/build# ./bin/main --help

usage: ./bin/main [options]

options:
  -h, --help            show this help message and exit
  --version             show version and build info
  -i, --interactive     run in interactive mode
  -p PROMPT, --prompt PROMPT
                        prompt to start generation with (default: empty)
  -f FNAME, --file FNAME
                        prompt file to start generation.
  -n N, --n-predict N   number of tokens to predict (default: -1, -1 = infinity, -2 = until context filled)
  -c N, --ctx-size N    size of the prompt context (default: 512, 0 = loaded from model)
  -b N, --batch-size N  logical maximum batch size (default: 2048)
```

Common - проблема интеграции llama.cpp

Конфигурация CMakeLists.txt содержит ограничения:

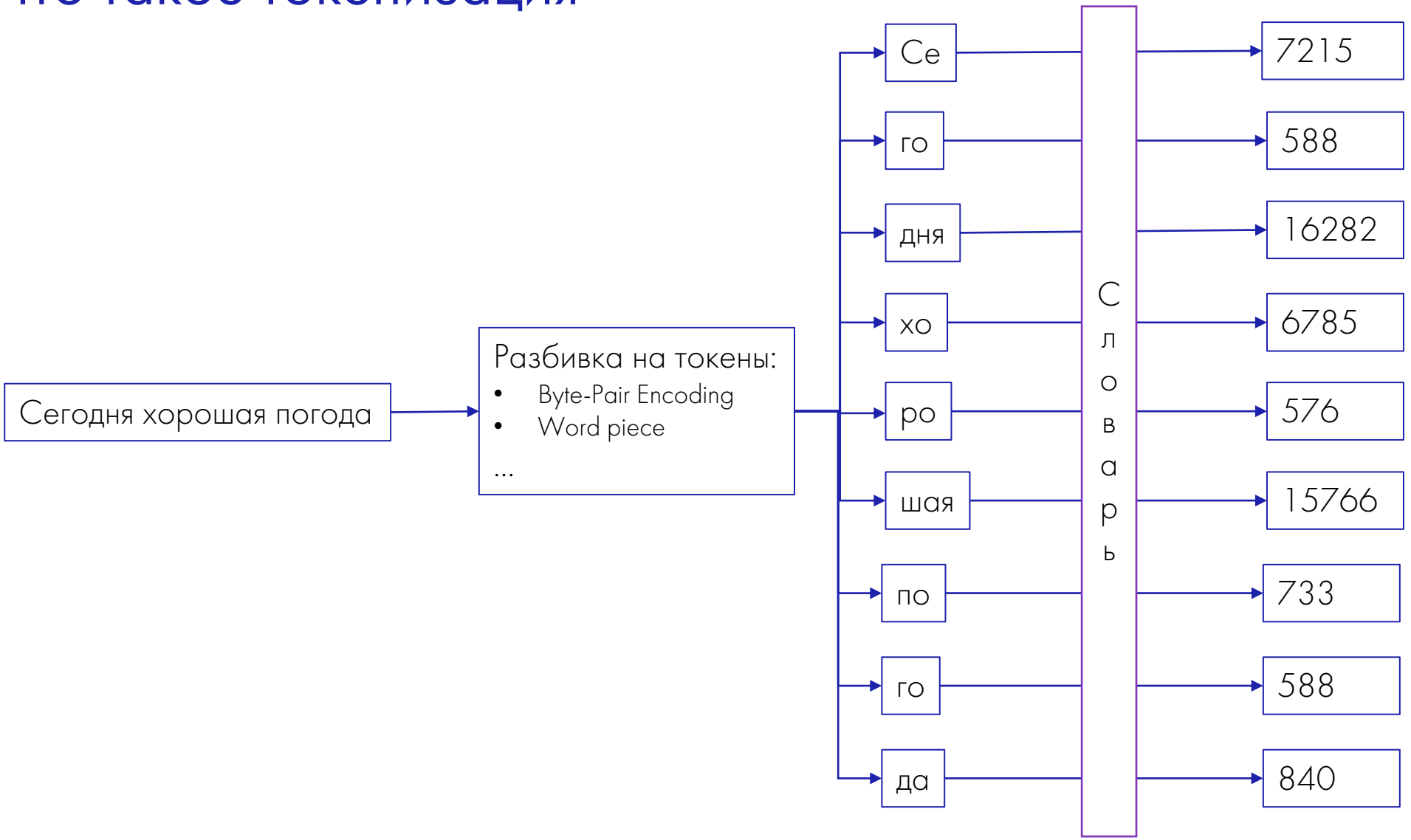
- Зависимости ориентированы на текущую директорию
  ```
  target_include_directories(${TARGET} PUBLIC .)
  ```
- Стандарт C++ ограничен 11й версией
  ```
  target_compile_features(${TARGET} PUBLIC cxx_std_11)
  ```
- Нет install инструкций

Решение – написать свой аналог нужных функций

Нам нужны функции для работы с :

1. Токенизацией
2. Сэмплированием
3. Авто-регрессией

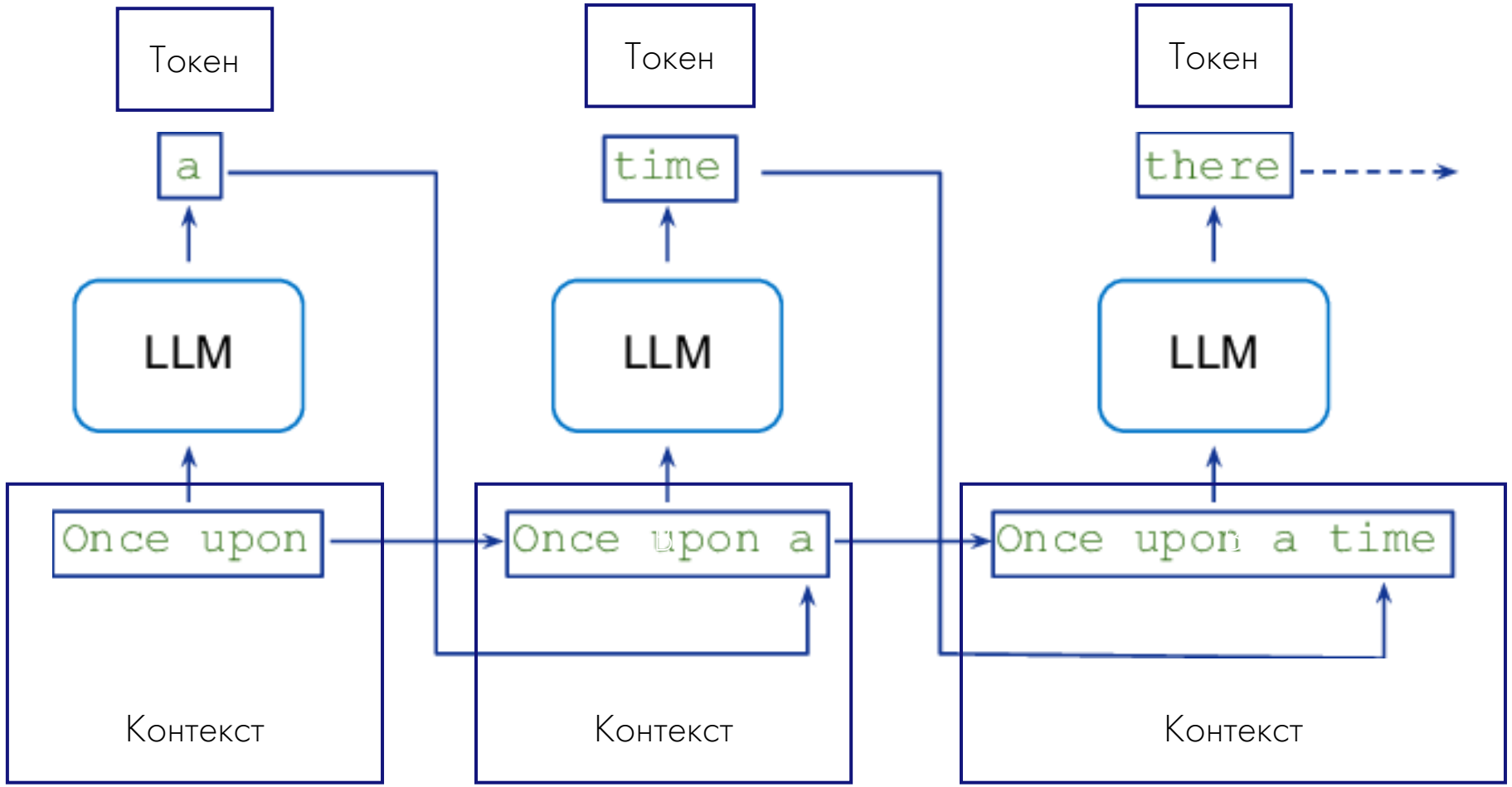
Что такое токенизация



Токенизация с llama.cpp

```
std::vector<llama_token> TextProcessor::tokenize(const std::string& text,
                                                 bool add_bos,
                                                 bool special) {
  int n_tokens = text.length() + add_bos;
  std::vector<llama_token> prompt_tokens(n_tokens);
  llama_tokenize(model_, text.data(), text.length(), prompt_tokens.data(),
                         prompt_tokens.size(), add_bos, special);
   …
   return prompt_tokens;
}
```

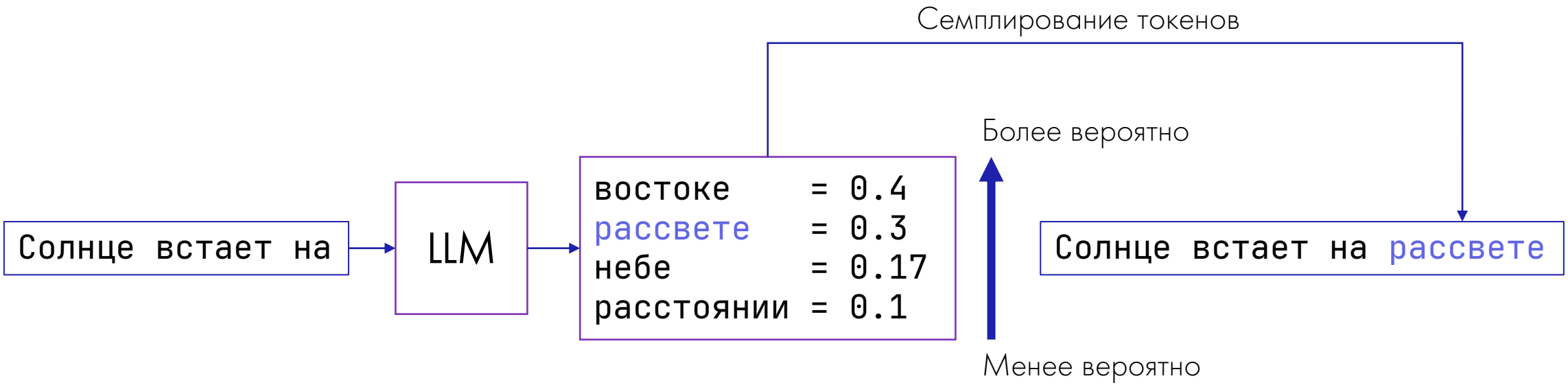
Что такое авторегрессия



Авто-регрессия с llama.cpp

```
void TextProcessor::predict(std::vector<llama_token>& embd, int& n_past) {
  …
// KV cache - для ускорения вычислений  и обработки длинного контекста
  llama_kv_cache_seq_rm(ctx_, 0, n_keep + 1, n_keep + n_discard + 1);
  llama_kv_cache_seq_shift(ctx_, 0, n_keep + 1 + n_discard, n_past, -n_discard);
  …
// результат декодирования сохраняется в ctx_
  llama_decode(ctx_, llama_batch_get_one(&embd[i], n_eval, n_past, 0)));
  …
  embd.clear();
}
```

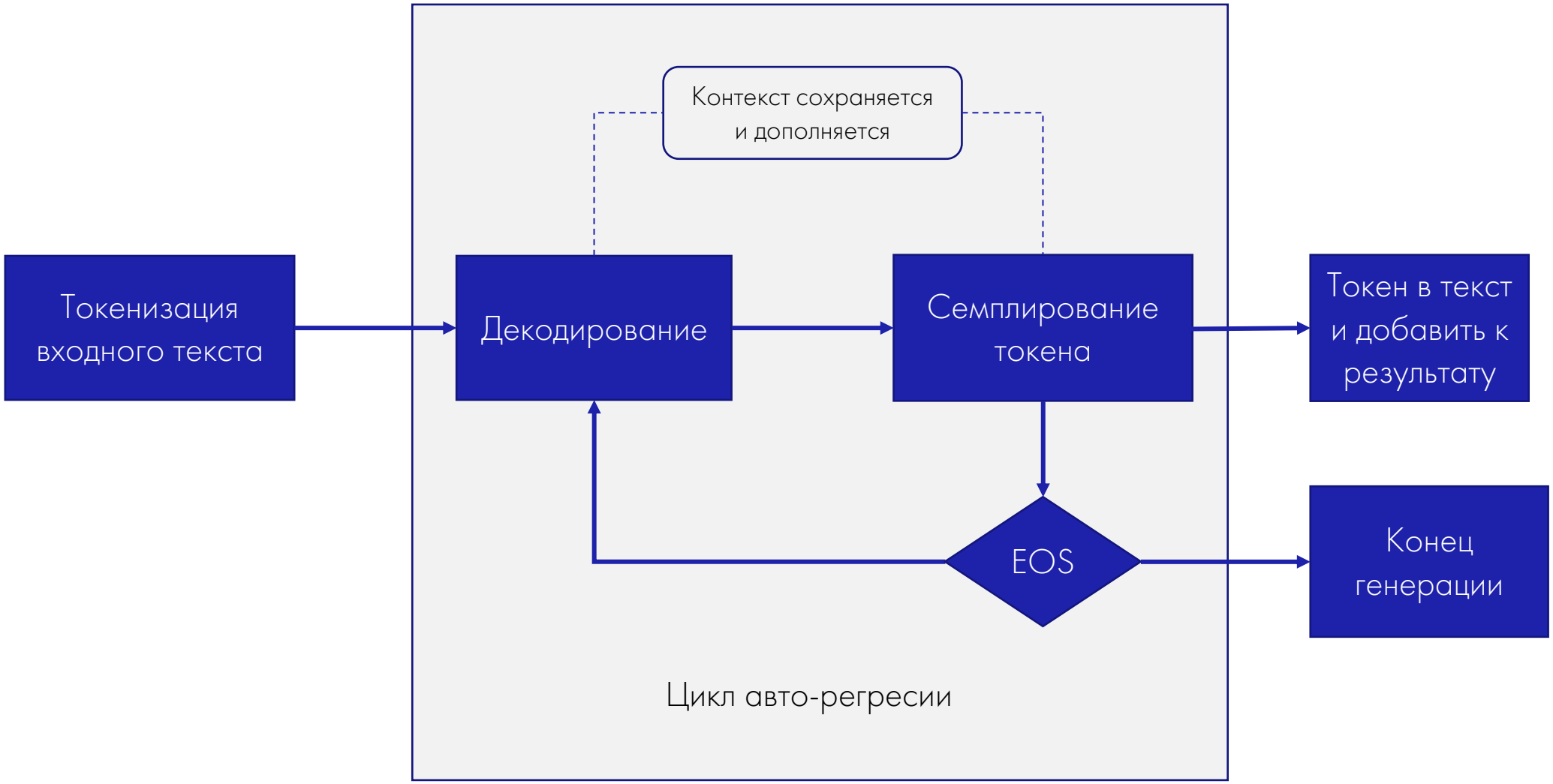
Что такое семплирование



Семплирование с llama.cpp

```
llama_token TextProcessor::sample_new_token(float temperature) {
…
  float* logits = llama_get_logits_ith(ctx_, idx);
  std::vector<llama_token_data> current_tokens_ <- [token_id, logits[token_id]}];
…
  llama_token_data_array cur_p = {current_tokens_};
…
  llama_sample_repetition_penalties(ctx_, &cur_p, prev_tokens_.data(),…);
…
  llama_sample_temp(ctx_, &cur_p, temperature);
  return llama_sample_token(ctx_, &cur_p);
}
```

Использование llama.cpp в своем приложении



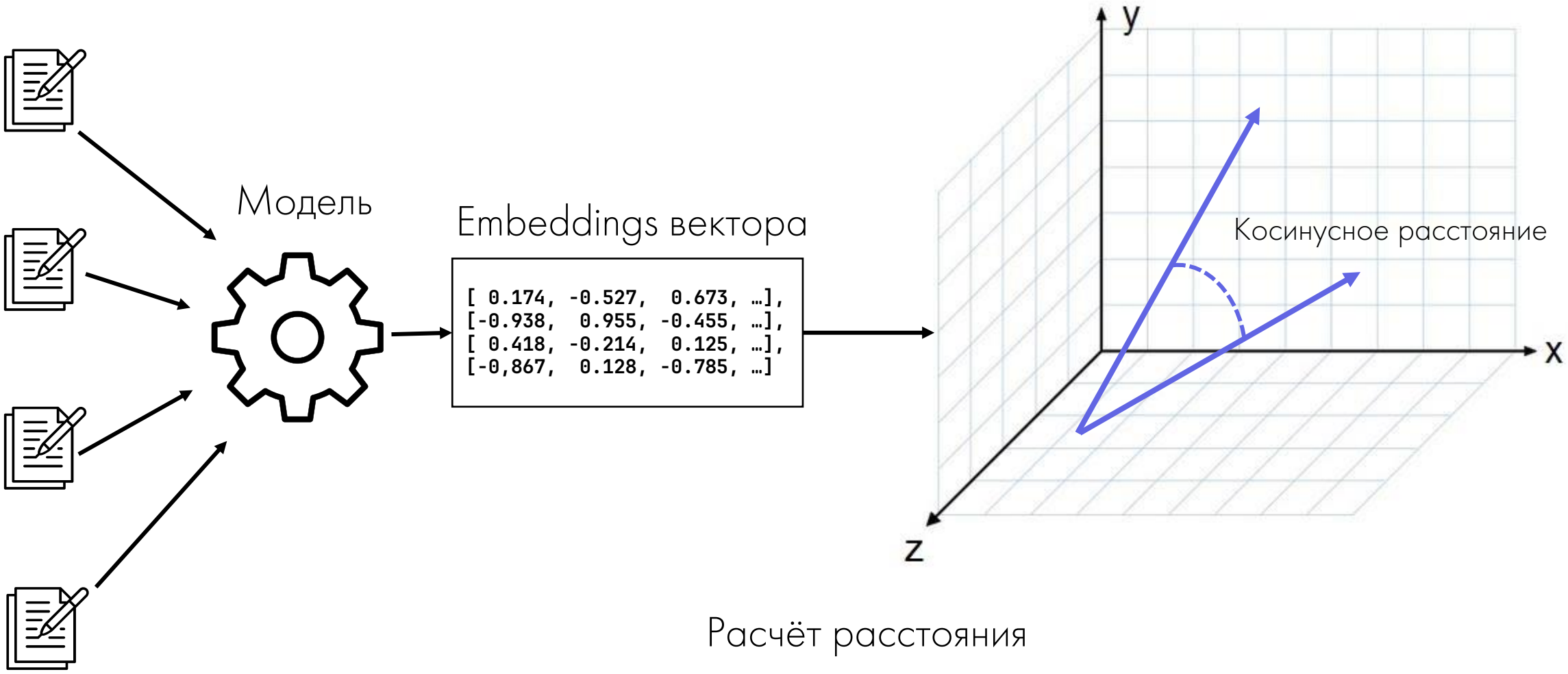
Как получить результат работы LLM

```
std::string TextProcessor::token_to_text(llama_token token) {
  std::vector<char>* buffer(8, 0);
  const int n_tokens =
      llama_token_to_piece(model_, token, buffer->data(), buffer->size());
  if (n_tokens < 0) {
    buffer->resize(-n_tokens);
    llama_token_to_piece(model_, token, buffer->data(), buffer->size());
  } else {
    buffer->resize(n_tokens);
  }
  return std::string(buffer->data(), buffer->size());
}
```

Итоги решения обобщения текстов и интеграции llama.cpp

- Модель LLaMa2 умеет понимать
- Saiga – для работы с русским
- Библиотеки GGML и llama подходят для интеграции
- Авторегрессию и токенизацию надо переписать самостоятельно
- TDLib – для работы с Telegram

Сравнение сообщений – Embeddings



Расчёт расстояния

Embeddings – из llama.cpp

```
ctx_params_ = llama_context_default_params();
ctx_params_.embedding = true;
ctx_ = llama_new_context_with_model(model_, ctx_params_);
llama_decode(ctx_, llama_batch_get_one(text_tokens.data(), n_tokens, n_past, 0));
const float* embeddings = llama_get_embeddings(ctx_);

const int n_embd = llama_n_embd(model_);
auto embed_vector =  dlib::mat(embeddings, n_embd);
```

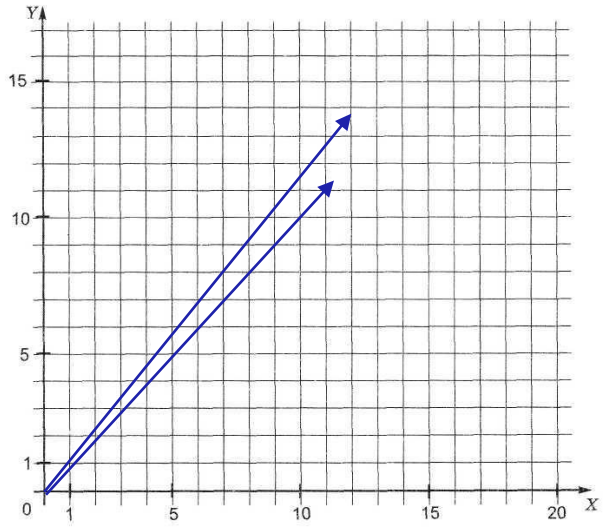
Проблема использования встроенных embeddings .

OpenELM: Новое семейство LM с открытым исходным кодом для обучения моделей и логического вывода.

[1,34,45,8]

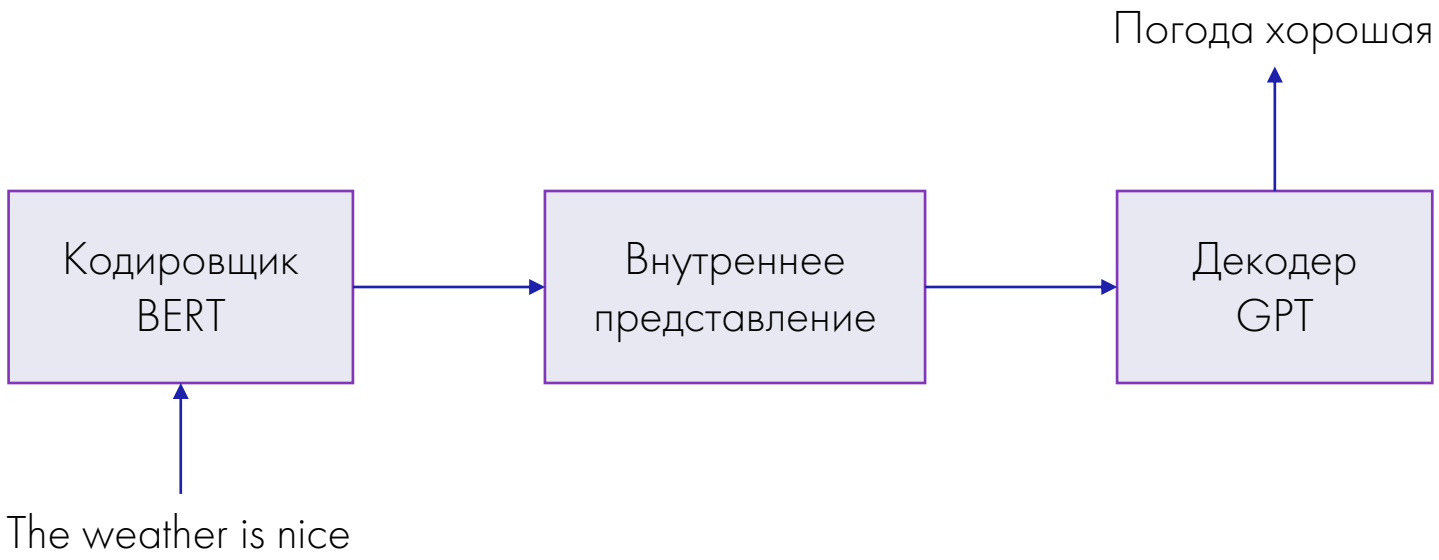
HiDiffusion: Новый метод, не требующий обучения. Повышает скорость предварительно обученных моделей.

[0.9,31,42,6]



Поиск решения получения эмбеддингов – BERT

Bidirectional Encoder Representations from Transformers



Готовая реализация получения эмбеддингов из текста – bert.cpp

- bert.cpp - запуск моделей Sentence BERT
- Эти модели ориентированы на создание качественных embeddings
- Реализован используя GGML без других зависимостей

Пример работы bert.cpp

a person is boiling soup.	a woman is placing eggs into a pan.	0.7
the man is slicing a potato.	a man is slicing potato.	0.01
a woman puts cosmetics on her eyelid.	the woman is penciling on eye shadow.	0.4
a cat is on a robot.	a man is eating bread.	0.8
a man jumping rope.	a man is talking.	0.5
a woman is putting on sun glasses.	a woman puts on sunglasses.	0.1

Проблемы интеграции bert.cpp

Версия GGML устарела

```
// было
struct ggml_cgraph gf = {};
// стало
struct ggml_cgraph* gf = ggml_new_graph_custom(ctx0, BERT_MAX_NODES, false);
...
ggml_graph_clear(gf);
```

```
// было
inpL = ggml_norm(ctx0, inpL);
// стало
inpL = ggml_norm(ctx0, inpL, model.hparams.f_norm_eps);
```

Снова проблема с русским языком

Трое мужчин играют в шахматы.	Двое мужчин играют в шахматы.	1
Человек курит.	Человек катается на коньках.	1
Человек играет на пианино.	Он играет на гитаре.	1
Мужчина играет на гитаре и поет.	Женщина играет на акустической гитаре и поет.	1
Женщина пишет.	Женщина плавает.	0.9
Человек играет на флейте.	Человек играет на бамбуковой флейте.	1
Человек складывает лист бумаги.	Кто-то складывает лист бумаги.	0.9
Человек бежит по дороге.	Панда-собака бежит по дороге.	0.9

Sentence RuBERT – для русского языка

Проблема с русским языком упорствует

C++ vs UTF-8

Решение проблемы с unicode – итераторы ICU

```
std::vector<std::string> words;

icu::UnicodeString icu_text =
icu::UnicodeString::fromUTF8(icu::StringPiece(text.c_str()));

icu::UnicodeString icu_norm_text;

icu::Normalizer::normalize(icu_text, UNormalizationMode::UNORM_DEFAULT, 0,
                           icu_norm_text, status);

icu::BreakIterator* bi = icu::BreakIterator::createWordInstance(icu::Locale("ru"),
                                                                status);

bi->setText(icu_norm_text);

…

delete bi;
```

Разбивка unicode текста на слова

```
while (current != icu::BreakIterator::DONE) {
  current = bi->next();
  auto s = bi->getRuleStatus();
  if (s == UBRK_WORD_NONE) {
    icu::UnicodeString icu_word(icu_norm_text, word_start, current - word_start);
    std::string word;
    icu_word.toUTF8String(word);
    words.push_back(word);
    word_start = current;
  }
}
```

Применение bert.cpp

```
auto bert_ctx_ = bert_load_from_file(model_file_path);
std::vector<bert_vocab_id> tokens_(bert_n_max_tokens(bert_ctx_));
bert_tokenize(bert_ctx_, text.c_str(), tokens_.data(),
              &num_tokens, max_tokens, config_->lang.c_str());
…
std::vector<float> embeddings_(bert_n_embd(bert_ctx_));
bert_eval(bert_ctx_, n_threads, tokens_.data(), tokens_.size(),
          embeddings_.data());
…
dlib::matrix<float> embd = dlib::mat(embeddings_);
…
dlib::cosine_distance distance_function;
auto bert_dist = distance_function(embd, embd_other);
```

Результаты после исправлений

UTF-8 + Sentence RuBERT от DeepPavlov

Трое мужчин играют в шахматы.	Двое мужчин играют в шахматы.	0.001
Человек курит.	Человек катается на коньках.	0.4
Человек играет на пианино.	Он играет на гитаре.	0.5
Мужчина играет на гитаре и поет.	Женщина играет на акустической гитаре и поет.	0.2
Женщина пишет.	Женщина плавает.	0.4
Человек играет на флейте.	Человек играет на бамбуковой флейте.	0.07
Человек складывает лист бумаги.	Кто-то складывает лист бумаги.	0.04
Человек бежит по дороге.	Панда-собака бежит по дороге.	0.2

Нельзя просто так взять

и использовать SentenceBERT для длинных текстов

Уменьшение количества предложений - через промпт

```
std::string summarize_instruction_ru(int n_predict,
                                     const std::string& text) const {
  return "\n\n### Задача:\n\n"
         "Напиши краткое содержание следующего текста."
         "\n\n### Текст:\n\n" + text +
         "\n\n### Максимальная длина ответа:\n\n" +std::to_string(n_predict) +" слов."
         "\n\n### Ответ:\n\n";
}
```

Пример обобщения текста – исходный текст

Оперируя биометрией в качестве средства авторизации, сервисы обязаны сдавать ее в Единую биометрическую систему (ЕБС). К этому их обязывает закон об идентификации с использованием биометрии (572-ФЗ), принятый Госдумой в конце 2022 г. В настоящее время сдавать биометрию россиян в ЕБС должны все организации, «владеющие системами, обеспечивающими аутентификацию на основе биометрии». Исключение составляют розыскные мероприятия, обеспечение безопасности государства, санитарно-эпидемиологического благополучия и пр.

Классифайды и сервисы каршеринга в список исключений не попали. Со слов источника издания, собираемая ими биометрия не отвечает требованиям ЕБС к фотографиям и образцам голоса. «Подключение к ЕБС и переведение цифровых фото в необходимый формат потребуют от компаний затрат, им проще будет отказаться от использования технологии», – добавил он.

Идея участников рабочей группы заключается в том чтобы перенести все сервисы, которые используют биометрию для борьбы с мошенничеством, в разряд исключений, и снять с них обязанность передавать биометрические данные россиян в ЕБС. В нынешних условиях, если эти сервисы не будут сдавать биометрию в ЕБС, это будет считаться нарушением закона, уточнил собеседник издания в российской ИТ-области.

165 слов

Разная длина обобщений

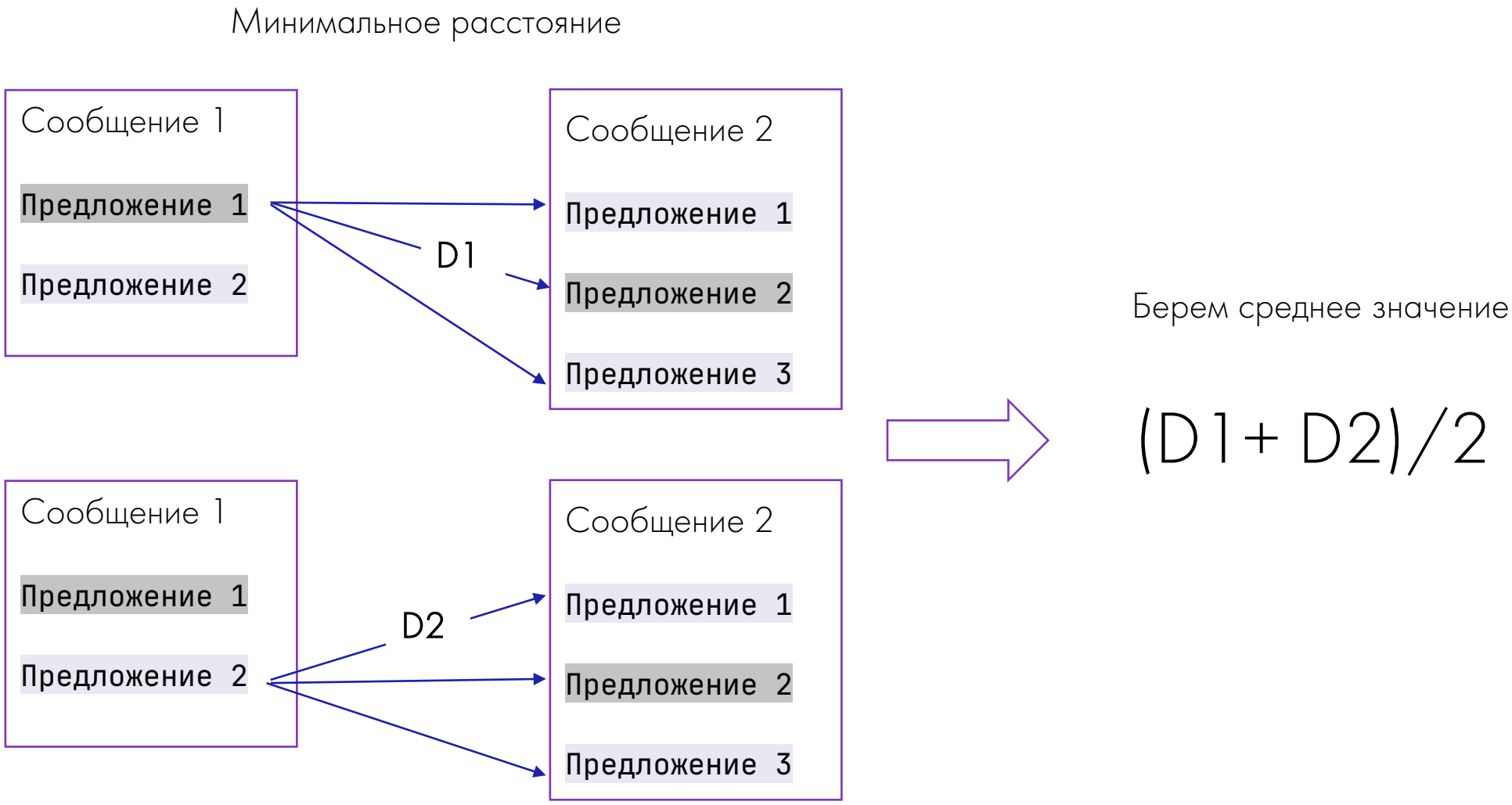
Сервисы каршеринга и классифайдов могут быть исключены из списка организаций, обязанных передавать биометрические данные россиян в Единую биометрическую систему (ЕБС). Это связано с тем, что их использование не отвечает требованиям к фотографиям и образцам голоса.

35 слов

Сервисы обязаны передавать биометрические данные россиян в Единую биометрическую систему, как это предусмотрено законом об идентификации с использованием биометрии. Однако некоторые организации могут быть исключены из этой обязанности, например, при проведении розыскных мероприятий или обеспечении безопасности государства. Участники рабочей группы предлагают перенести все сервисы, использующие биометрию для борьбы с мошенничеством, в разряд исключений и освободить их от необходимости передавать данные в ЕБС, чтобы избежать дополнительных затрат на адаптацию технологии.

69 слов

Как же все-таки сравнивать несколько предложений?



Сравнение эмбеддингов сообщений

```cpp
double embeddings_distance(const EmbeddingsBag& a, const EmbeddingsBag& b){
  // сортируем a и b по длине получая small и big наборы соответственно
  auto k = small->size();
  dlib::cosine_distance distance_function;
  auto result_dist = oneapi::tbb::parallel_reduce(
      oneapi::tbb::blocked_range<size_t>(0, k), 0.0,
        // вычисляем сумму минимальных расстояний до противоположных embeddings
      },
      [](auto dist_a, auto dist_b) { return dist_a + dist_b; });
  return result_dist / k; // среднее значение минимального расстояния
```

Сумма минимальных расстояний

```
"""
[&](const auto& range, auto partial_dist) {
  for (std::size_t i = range.begin(); i != range.end(); ++i) {
    double distance = std::numeric_limits<double>::max();
    for (const auto& s : *big) {
      // сравниваем расстояние от элемента small к элементам big
      distance = std::min(distance, distance_function((*small)[i], s));
    }
    partial_dist += distance;
  }
  return partial_dist;
},
"""
```

Итоги решения задачи сравнения сообщений

- Использовать BERT для эмбеддингов
- Sentence RuBert от DeepPavlov для русского
- bert.cpp – для запуска сети
- Руками допилить совместимость GGML
- Решить проблему с UTF-8
- Сеть из первой задачи для создания коротких обобщений
- Для сравнения нескольких предложений нужен свой алгоритм сравнения

Алгоритм получения дайджеста

1. Получить список сообщений
2. Каждое сообщение максимально обобщить
3. Для каждого обобщения получить эмбеддинги для каждого предложения
4. Для всех пар обобщений посчитать расстояния
5. Разделить обобщения на кластера с похожими расстояниями между собой
6. Выбрать из каждого кластера по одному сообщению

Конфигурация приложения

Модели:

- Saiga - 13 миллиардов параметров
- ruBert - 180 миллионов параметров

Каналы :

- ё-Пром | Импортозамещение в промышленности
- Новости Промышленности
- Время - вперёд!
- Максим imaxai Горшенин
- RUSmicro

Пример похожих сообщений

Группа «Россети» перешла на российскую операционную систему. «Базальт СПО» поставила «Россетям» свыше 97 000 лицензий «Альт рабочая станция» для компьютеров и ноутбуков и 3000 лицензий «Альт сервер» для серверов различного назначения.

Расстояние = 0.07

Компания «Базальт СПО» поставляет 100 тысяч лицензий на операционные системы для компьютеров и серверов «Россети». ОС «Альт» включена в реестр российского ПО, перевод офисного контура на эту ОС. Компания помогает предприятиям перейти с зарубежных систем на российские.

Пример похожих сообщений

«Прогресс МС-26» успешно стартовал и направляется к **МКС**, в пути будет 2 суток, на борту 2518 кг **грузов** для станции.

Расстояние = 0.056

Ракета-носитель «Союз-2.1а» выводит грузовой космический корабль **"Прогресс МС-26"** на орбиту, а через два дня он должен доставить **грузы** к **Международной космической станции**.

Пример отличающихся сообщений

Ракета-носитель «Союз-2.1а» стартовала, на борту транспортный грузовой корабль "Прогресс МС-26". Через девять минут отделения от третьей ступени и через двое суток - прибытие к МКС.

«Прогресс МС-26» успешно стартовал и направляется к МКС, в пути будет 2 суток, на борту 2518 кг грузов для станции.

Расстояние = 0.5

Интересный пример сравнения

Ракета-носитель «Союз-2.1а» стартовала, на борту транспортный грузовой корабль "Прогресс МС-26". Через девять минут отделения от третьей ступени и через двое суток - прибытие к МКС.

Расстояние = 0.34 ?

Пожар в гигантском деревянном ангаре, который использовался при съемках сериалов Секретные материалы и Звездный путь, удалось потушить спустя 24 дня после возгорания.

Какая производительность ?

Система	Модель	Tokens Per Second
Intel Core i9-13900HX RAM 32GiB NVIDIA GeForce RTX 4070 8GiB	LlaMa 2 / 13b / 4 bits	~7
Qualcomm Kryo-4XX-Silver(POCO 5G) RAM 8GiB	LlaMa 2 / 4b / 4 bits	~5-6

Сколько нужно памяти?

Количество параметров/ Размер типа данных	7 миллиардов	13 миллиардов	70 миллиардов
32 bits float	33.6 GiB	62.1 GiB	335 GiB
16 bits float	16.8 GiB	31.2 GiB	168 GiB
8 bits int	8.4 GiB	15.6 GiB	84 GiB
4 bits int	4.2 GiB	7.8 GiB	42 GiB

Что бы запустить квантованную до 16 бит модель понадобится 2 карты A100 80GB

Какие выводы для себя я сделал

- Используя библиотеки на базе llama.cpp можно эффективно решать задачи обработки текстов
- Определенный круг задач NLP можно решать как на домашних ПК так и телефонах
- Если что-то не работает не надо бросать инструмент, а можно попытаться разобраться
- LLM позволяют решать большой круг задач без доработки через промты
- Но для эффективного решения лучше использовать специализированные инструменты

Мое решение работает но сейчас я бы делал иначе

- Поддержка BERT уже встроена в llama.cpp
- Можно сделать fine-tuning модели Saiga под задачу обобщения
- Реализовать обобщение и сравнение в рамках моделей на базе BERT
- Для работы эмбеддингами использовать векторную БД

Ссылки на материалы

1. Код примера
   https://gitflic.ru/project/kolkir/llama_td_analyser
2. llama.cpp
   https://github.com/ggerganov/llama.cpp/tree/master
3. bert.cpp
   https://github.com/skeskinen/bert.cpp
4. Sentence RuBert от DeepPavlov
   https://huggingface.co/DeepPavlov/rubert-base-cased-sentence
5. Модели Saiga
   https://huggingface.co/collections/IlyaGusev/
6. ICU
   https://icu.unicode.org/
7. TDLib
   https://github.com/tdlib/td