

Языковые модели

Language models

Языковая модель

Языковая модель оценивает **вероятности** разных **токенов**:


- слов
- букв
- ...

или **вероятности последовательностей токенов**:

- предложений
- слов
- ...

Языковая модель



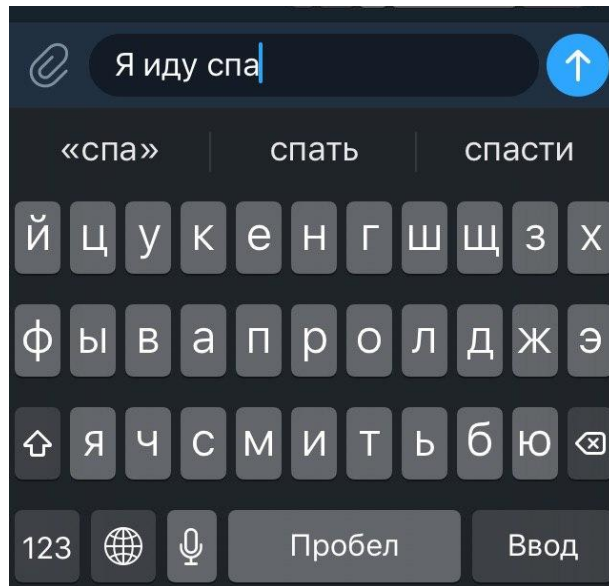
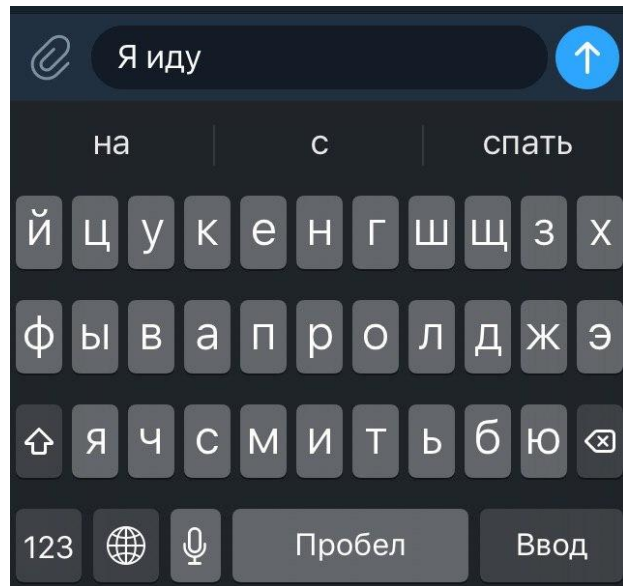
a lazy dog | 

a lazy dog **breed**
a lazy dog **jump over**
a lazy dog **jumps over the fox**
a lazy dog **restaurant**
a lazy dog
the lazy dog **menu**
the lazy dog **cafe**
the lazy dog **cookie co**
the lazy dog **boracay**
the lazy dog **roseville**

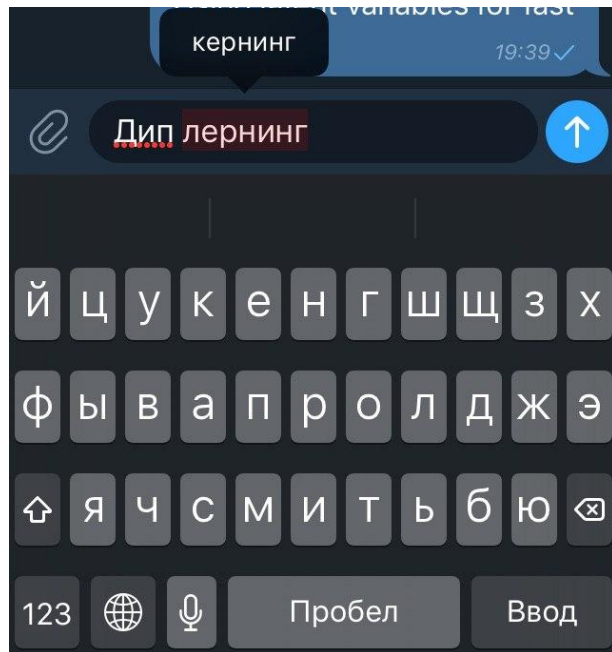
[Поиск в Google](#) [Мне повезёт!](#)

[Пожаловаться на неприемлемые подсказки](#)

Языковая модель

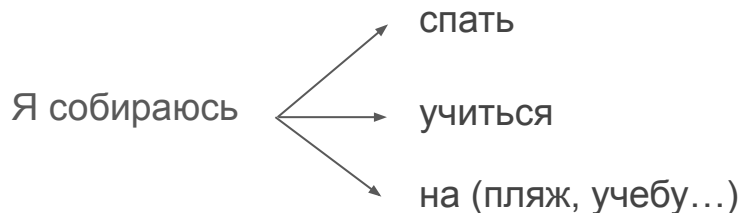


Языковая модель



Языковая модель

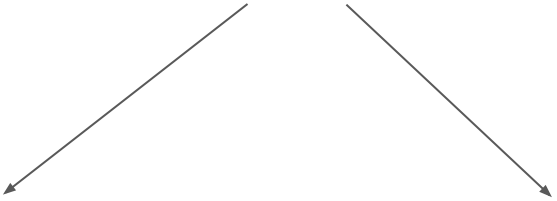
Вероятность токена: насколько вероятно, что данный токен появится после заданной последовательности токенов



Вероятность предложения: насколько вероятно, что данное предложение встретится в естественном языке

$$P(\text{Я собираюсь на пляж}) > P(\text{Пляж собираюсь на я})$$

Языковые модели



```
graph TD; A[Языковые модели] --> B[Частотные (Count-based)]; A --> C[Нейронные (Neural)];
```

Частотные
(Count-based)

Нейронные
(Neural)

Count-based LM

Оценка вероятности

Давайте посчитаем количество конфеток Бerti-боттс каждого вкуса в пачке.

Банан	Черный перец	Корица	Лимон	Грязь	Дождевой червь	Трава
26	23	19	20	17	13	22

$$P(taste) = \frac{Count(taste)}{\sum Count(taste)}$$

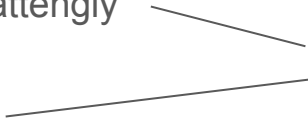
Оценка вероятности: предложения

- хочется оценивать вероятности появления предложений в естественном языке, используя ту же идею.
- у нас есть датасет с большим количеством предложений
- также мы хотим уметь оценивать вероятность появления предложения, которое мы **никогда не видели раньше**.

The Ftaghn credled raftangly spattengly

Ftagn ranght kdufbfnms msnlkvrl

$P = ?$



Оценка вероятности: предложения

Проблемы:

- у нас всегда мало данных
- и всегда много предложений, которые ни разу не встречаются в корпусе данных

Оценка вероятности: предложения

Проблемы:

- у нас всегда мало данных
- и всегда много предложений, которые ни разу не встречаются в корпусе данных

Решение:

- оценивать вероятность предложения как комбинацию вероятностей его частей: **N-gram Language model**

Оценка вероятности: предложения

Хотим получить $P(\text{Я хочу спать})$

Идея:

$$1. \quad P(\text{Я хочу спать}) = P(w_1 = \text{Я}, w_2 = \text{хочу}, w_3 = \text{спать})$$

$$2. \quad P(X, Y) = P(Y | X) P(X)$$

$$P(\text{Я хочу спать}) = P(\text{спать} | \text{Я хочу}) \cdot P(\text{Я хочу}) =$$

$$= P(\text{спать} | \text{Я хочу}) \cdot P(\text{хочу} | \text{Я}) \cdot P(\text{Я})$$

Оценка вероятности: предложения

$$P(X, Y) = P(Y | X) P(X)$$

$$P(\text{Я хочу спать}) = P(\text{спать} | \text{Я хочу}) \cdot P(\text{Я хочу}) =$$

$$= P(\text{спать} | \text{Я хочу}) \cdot P(\text{хочу} | \text{Я}) \cdot P(\text{Я})$$

$$P(\text{спать} | \text{Я хочу}) = \frac{\text{Count}(\text{Я хочу спать})}{\text{Count}(\text{Я хочу})}$$

Оценка вероятности: предложения

$$\begin{aligned} P(\text{Я хочу спать}) &= P(\text{спать} \mid \text{Я хочу}) \cdot P(\text{Я хочу}) = \\ &= P(\text{спать} \mid \text{Я хочу}) \cdot P(\text{хочу} \mid \text{Я}) \cdot P(\text{Я}) \end{aligned}$$

Проблема:

нам все еще нужно уметь оценивать вероятности огромных кусков предложений:

$$\begin{aligned} P(\text{Милая овечка наслаждается беззаботной жизнью}) &= \\ P(\text{жизнью} \mid \text{Милая овечка наслаждается беззаботной}) \cdot \dots \end{aligned}$$

||

$$\frac{\text{Count}(\text{Милая овечка наслаждается беззаботной жизнью})}{\text{Count}(\text{Милая овечка наслаждается беззаботной})}$$

Оценка вероятности: предложения

Решение: **допущение независимости** (independence assumption):

следующее слово зависит только от n предыдущих:

- **trigram model:** $P(w_i \mid w_1, w_2, \dots, w_{i-1}) \sim P(w_i \mid w_{i-1}, w_{i-2})$
- **bigram model:** $P(w_i \mid w_1, w_2, \dots, w_{i-1}) \sim P(w_i \mid w_{i-1})$
- **unigram model:** $P(w_i \mid w_1, w_2, \dots, w_{i-1}) \sim P(w_i)$

Оценка вероятности: предложения

Итоговая оценка вероятности:

$$P(w_i | w_1, \dots, w_{i-1}) = P(w_i | w_{i-k}, \dots, w_{i-1}) = \frac{Count(w_i, w_{i-1}, \dots, w_{i-k})}{Count(w_{i-1}, \dots, w_{i-k})}$$

$$P(\text{спать} | \text{Я очень хочу лечь}) = P(\text{спать} | \text{хочу лечь}) = \frac{Count(\text{хочу лечь спать})}{Count(\text{хочу лечь})}$$

Оценка вероятности: предложения

Trigram language model:

$P(\text{Милая овечка наслаждается беззаботной жизнью}) =$

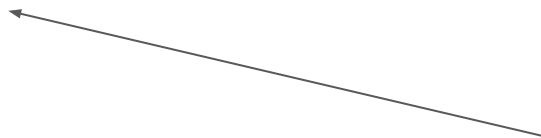
$= P(\text{жизнью} \mid \text{наслаждается беззаботной}) \cdot P(\text{беззаботной} \mid \text{овечка наслаждается}) \cdot P(\text{наслаждается} \mid \text{милая овечка}) \cdot P(\text{овечка} \mid \text{милая}) \cdot P(\text{милая})$

Оценка вероятности: предложения

Trigram language model:

$P(\text{Милая овечка наслаждается беззаботной жизнью}) =$

$= P(\text{жизнью} \mid \text{наслаждается беззаботной}) \cdot P(\text{беззаботной} \mid \text{овечка наслаждается}) \cdot P(\text{наслаждается} \mid \text{милая овечка}) \cdot P(\text{овечка} \mid \text{милая}) \cdot P(\text{милая})$



Проблема: что если мы никогда не видели в датасете такой префикс?

Оценка вероятности: предложения

Trigram language model:

$P(\text{Милая овечка наслаждается беззаботной жизнью}) =$

$= P(\text{жизнью} \mid \text{наслаждается беззаботной}) \cdot P(\text{беззаботной} \mid \text{овечка наслаждается}) \cdot P(\text{наслаждается} \mid \text{милая овечка}) \cdot P(\text{овечка} \mid \text{милая}) \cdot P(\text{милая})$



Проблема: что если мы никогда не видели в датасете такой префикс?

Решение: **backoff**: $\frac{Count(\text{Милая овечка наслаждается})}{Count(\text{Милая овечка})} = \frac{Count(\text{овечка наслаждается})}{Count(\text{овечка})}$

Языковая модель

Как построить триграмную языковую модель: $P(w_i | w_1, w_2, \dots, w_{i-1}) \sim P(w_i | w_{i-1}, w_{i-2})$

- Посчитать частоты встречаемостей всех **слов** в датасете, всех **пар слов** и всех **троек слов** в датасете
- Для оценки вероятности предложения использовать формулу полной вероятности $P(X, Y) = P(Y | X) P(X)$

$P(\text{Милая овечка наслаждается беззаботной жизнью}) =$

$= P(\text{жизнью} | \text{наслаждается беззаботной}) \cdot P(\text{беззаботной} | \text{овечка наслаждается})$
 $\cdot P(\text{наслаждается} | \text{милая овечка}) \cdot P(\text{овечка} | \text{милая}) \cdot P(\text{милая})$

- Вычислить условные вероятности, используя подсчитанные частоты встречаемости слов и пар слов :

$$P(\text{наслаждается} | \text{Милая овечка}) = \frac{\text{Count}(\text{Милая овечка наслаждается})}{\text{Count}(\text{Милая овечка})}$$

Оценка вероятности токена

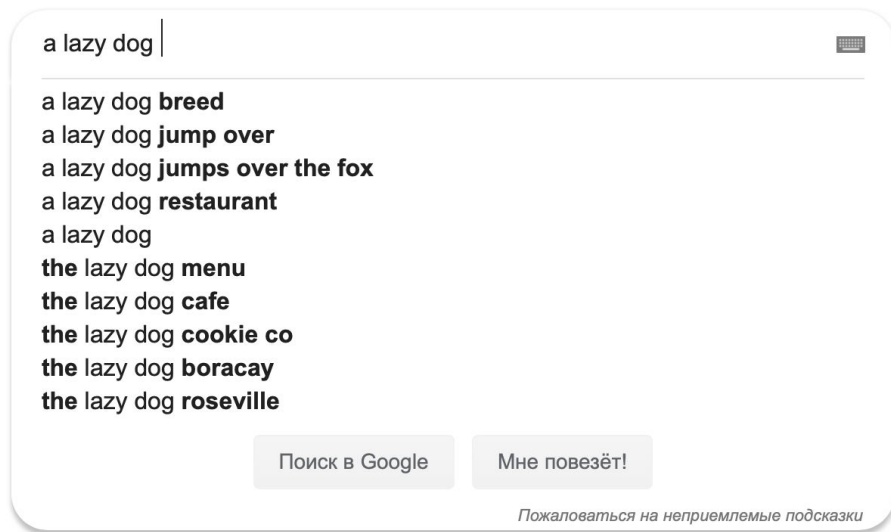
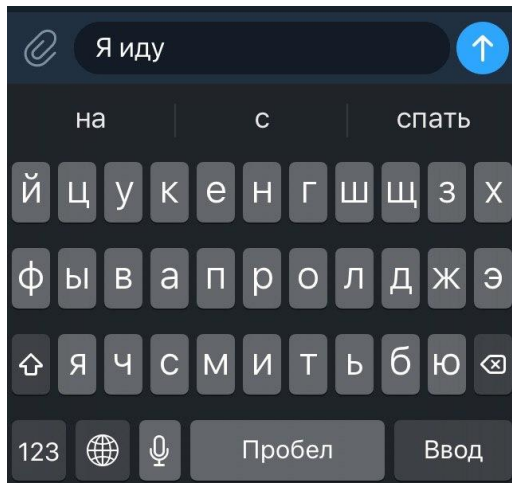
~~Рыжая лиса очень быстро~~ несется в ____
└──────────┘



$$P(w_i | \text{Рыжая лиса очень быстро несется в } \dots) = P(w_i | \text{быстро несется в } \dots) = \frac{\text{Count}(\text{несется в } w_i)}{\text{Count}(\text{несется в})}$$

Оценка вероятности токена

Теперь мы знаем, как
сделать подсказки к запросам
в Google и T9 =)



Генерация текста языковой моделью

Милая __

Генерация текста языковой моделью

Милая _



кошка	0.35
собака	0.3
девушка	0.15
лиса	0.05
часы	0.0006
лук	0.00004
идти	0.00000000012

$\text{Count}(X \mid \text{Милая})$

$\text{Count}(\text{Милая})$

Генерация текста языковой моделью

Милая кошка

Генерация текста языковой моделью

Милая кошка _____



лежит	0.44
сидит	0.12
бежит	0.06
бежать	0.0002
стоит	0.0006
лук	0.000009
идти	0.000000000012

$\text{Count}(X \mid \text{кошка})$

$\text{Count}(\text{кошка})$

Генерация текста языковой моделью

Милая кошка лежит —

Оценка качества языковых моделей

1. Вероятность / perplexity

оценить вероятности предложений из тестового датасета с точки зрения нашей LM.

Чем вероятности больше, тем модель лучше.

Perplexity: $PP(W) = \sqrt[N]{\frac{1}{P(w_1, w_2, \dots, w_N)}}$ Чем perplexity меньше, тем лучше.

2. Внедрить LM в систему NLP и посмотреть, какая из LM работает лучше.

Языковые модели

Есть много улучшений count-based языковых моделей, которые влияют на подсчет вероятности возникновения токена $P(w_i | w_1, \dots, w_{i-1})$

- сглаживание Лапласа
- сглаживание Гуда-Тьюринга,
- сглаживание Виттена-Белла
- сглаживание Кнезера-Нея (Kneser-Ney)
- ...

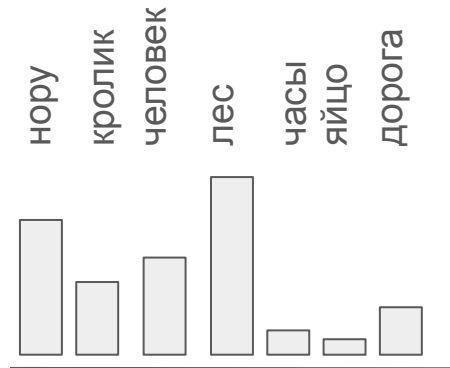
$$p_{KN}(w_i | w_{i-n+1}^{i-1}) = \frac{\max(c(w_{i-n+1}^{i-1}, w_i) - \delta, 0)}{\sum_{w'} c(w_{i-n+1}^{i-1}, w')} + \delta \frac{|\{w' : 0 < c(w_{i-n+1}^{i-1}, w')\}|}{\sum_{w'} c(w_{i-n+1}^{i-1}, w')} p_{KN}(w_i | w_{i-n+2}^{i-1})$$

Neural LM

Нейронные языковые модели

Neural LM

(1.000.000,)



Распределение
вероятностей

(1536,)



FC

(512,)

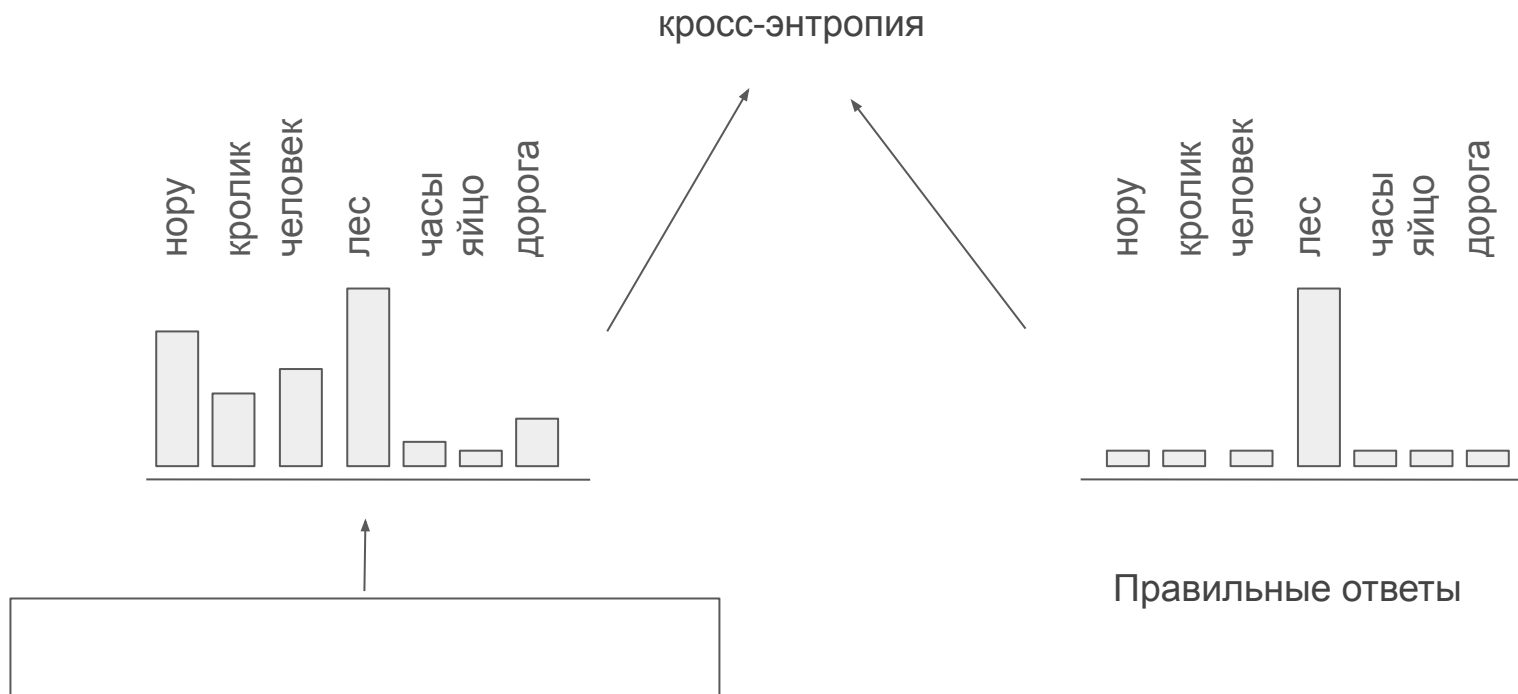


Embeddings
(напр. конкатенация)

~~Рыжая лиса очень~~

быстро несется в

Neural LM



Neural LM

Преимущества:

- нет проблемы со словами, которые не встречались в датасете
- нет проблемы с нулевыми значениями

Neural LM

Преимущества:

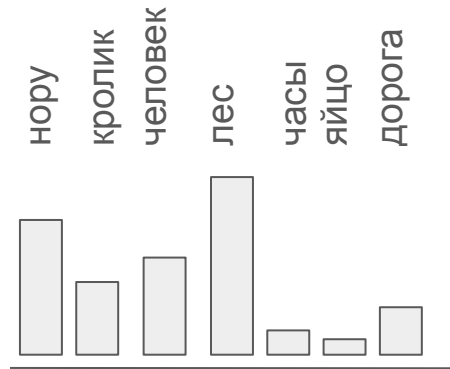
- нет проблемы со словами, которые не встречались в датасете
- нет проблемы с нулевыми значениями

Оставшиеся проблемы:

- марковское свойство

Neural LM

(1.000.000,)



Распределение вероятностей

(?,)



FC

(?,)

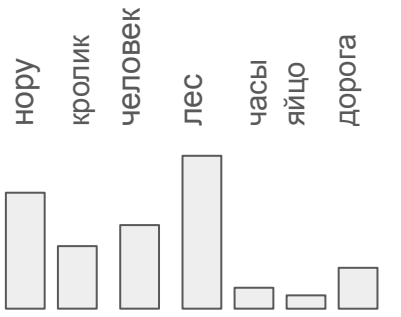


RNN

<bos> Рыжая лиса очень быстро несется в

Neural LM

(1.000.000,)



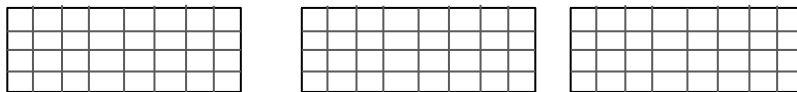
Распределение вероятностей



FC

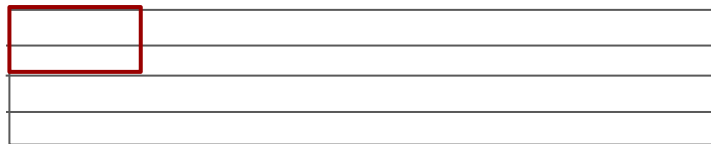


Flatten



Карты активации

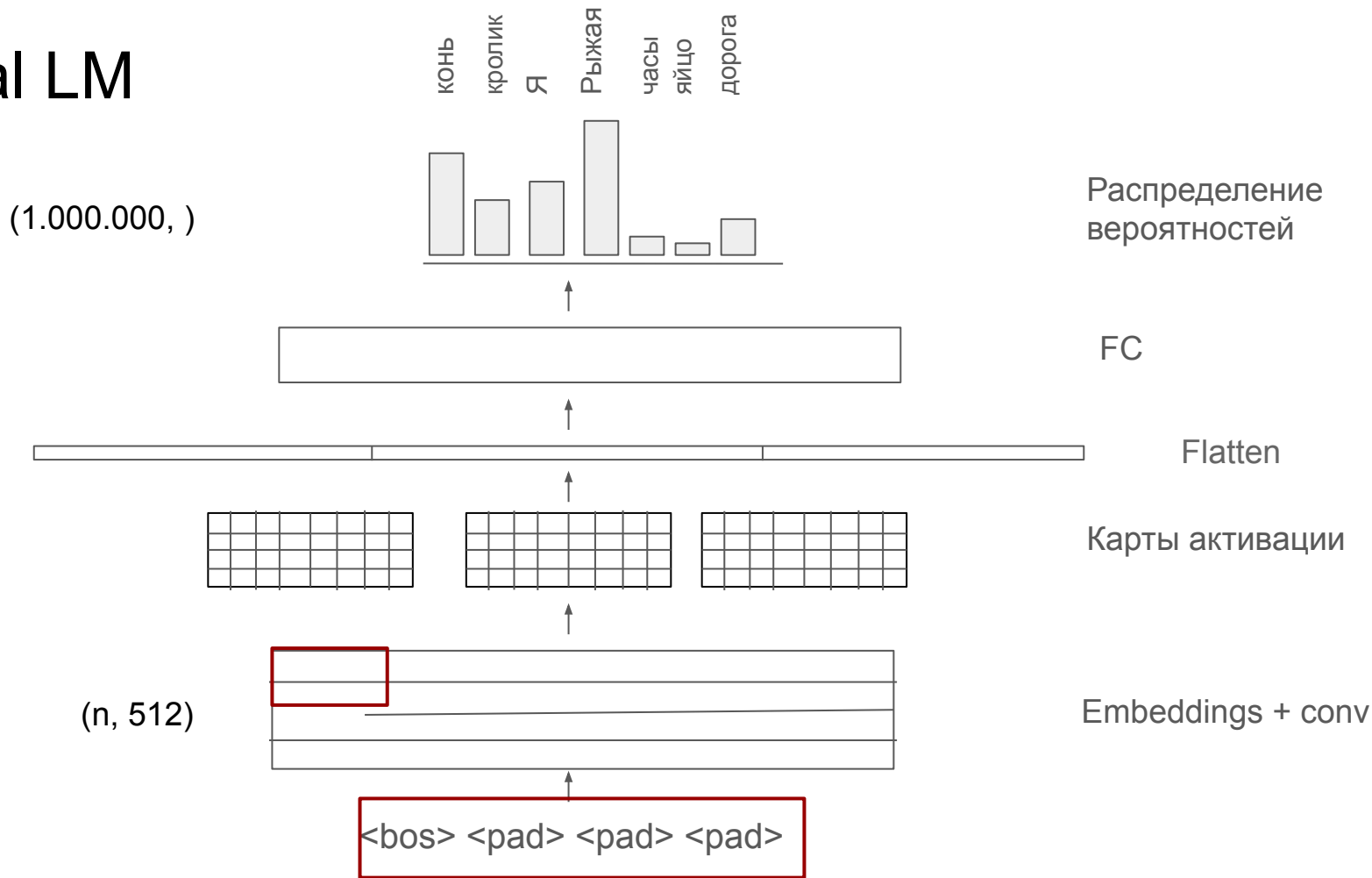
(n, 512)



Embeddings + conv

<bos> <pad> <pad> <pad> Рыжая лиса очень быстро несется в

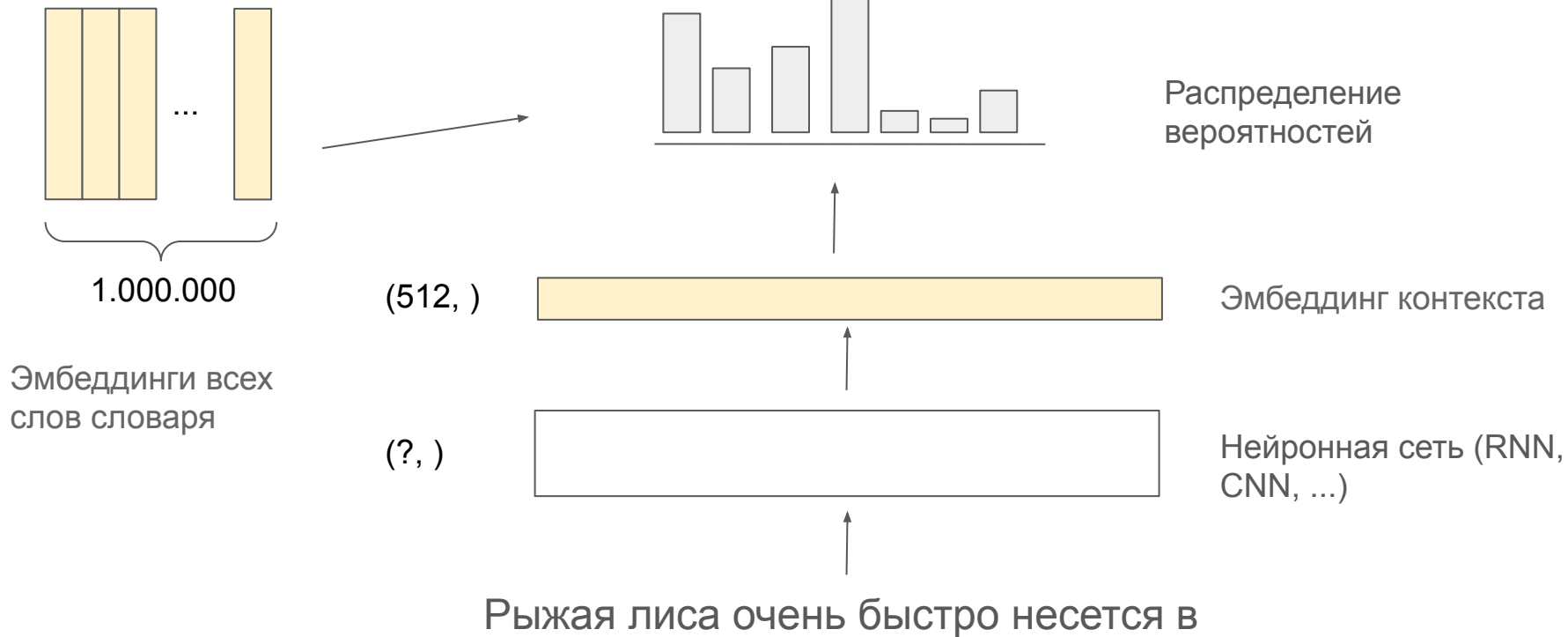
Neural LM



Сверточные Neural NM

- можно комбинировать несколько сверточных слоев
- можно комбинировать CNN + RNN
- не используют pooling
- можно использовать skip connection
- предсказывают следующее слово по префиксу фиксированной длины, но этот префикс может быть довольно длинным

Neural LM



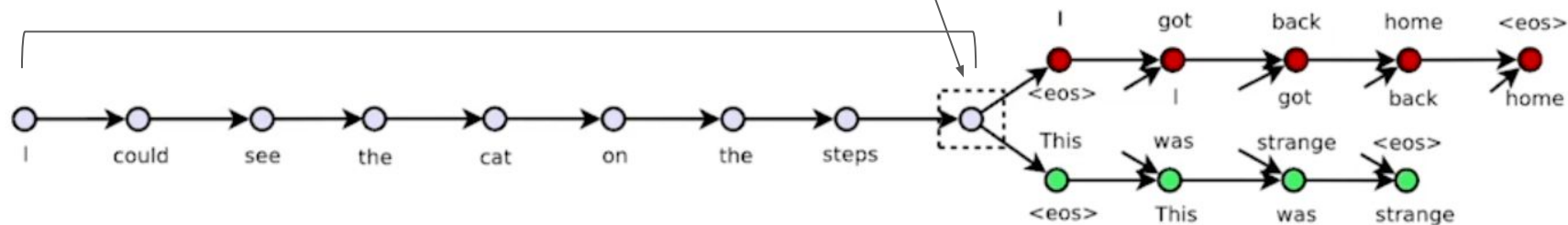
Эмбе́ддинги из языковой модели

С помощью обученной языковой модели можно получать эмбе́ддинги предложений

Эмбединги из языковой модели

Генерация следующего предложения

RNN



Примеры работы языковых моделей

For $\bigoplus_{n=1, \dots, m} \mathcal{L}_{m, \bullet} = 0$, hence we can find a closed subset \mathcal{H} in \mathcal{H} and any sets \mathcal{F} on X , U is a closed immersion of S , then $U \rightarrow T$ is a separated algebraic space.

Proof. Proof of (1). It also start we get

$$S = \mathrm{Spec}(R) = U \times_X U \times_X U$$

and the comparicoly in the fibre product covering we have to prove the lemma generated by $\coprod Z \times_U U \rightarrow V$. Consider the maps M along the set of points Sch_{fppf} and $U \rightarrow U$ is the fibre category of S in U in Section, ?? and the fact that any U affine, see Morphisms, Lemma ?? . Hence we obtain a scheme S and any open subset $W \subset U$ in $\mathrm{Sh}(G)$ such that $\mathrm{Spec}(R') \rightarrow S$ is smooth or an

$$U = \bigcup U_i \times_{S_i} U_i$$

which has a nonzero morphism we may assume that f_i is of finite presentation over S . We claim that $\mathcal{O}_{X, x}$ is a scheme where $x, x', s'' \in S'$ such that $\mathcal{O}_{X, x'} \rightarrow \mathcal{O}'_{X', x'}$ is separated. By Algebra, Lemma ?? we can define a map of complexes $\mathrm{GL}_{S'}(x'/S'')$ and we win. \square

To prove study we see that \mathcal{F}_U is a covering of \mathcal{X}' , and \mathcal{T}_i is an object of $\mathcal{F}_{X/S}$ for $i > 0$ and \mathcal{F}_p exists and let \mathcal{F}_i be a presheaf of \mathcal{O}_X -modules on \mathcal{C} as a \mathcal{F} -module. In particular $\mathcal{F} = U/\mathcal{F}$ we have to show that

$$\widetilde{M}^\bullet = \mathcal{I}^\bullet \otimes_{\mathrm{Spec}(k)} \mathcal{O}_{S, s} - i_X^{-1} \mathcal{F}$$

is a unique morphism of algebraic stacks. Note that

$$\mathrm{Arrows} = (\mathrm{Sch}/S)_{fppf}^{\mathrm{opp}}, (\mathrm{Sch}/S)_{fppf}$$

and

$$V = \Gamma(S, \mathcal{O}) \mapsto (U, \mathrm{Spec}(A))$$

is an open subset of X . Thus U is affine. This is a continuous map of X is the inverse, the groupoid scheme S .

Proof. See discussion of sheaves of sets. \square

The result for prove any open covering follows from the less of Example ?? . It may replace S by $X_{\mathrm{spaces}, \text{étale}}$ which gives an open subspace of X and T equal to S_{zar} , see Descent, Lemma ?? . Namely, by Lemma ?? we see that R is geometrically regular over S .

Lemma 0.1. Assume (3) and (3) by the construction in the description.

Suppose $X = \lim |X|$ (by the formal open covering X and a single map $\mathrm{Proj}_X(\mathcal{A}) = \mathrm{Spec}(B)$ over U compatible with the complex

$$\mathrm{Set}(\mathcal{A}) = \Gamma(X, \mathcal{O}_{X, \mathcal{O}_X}).$$

When in this case of to show that $\mathcal{Q} \rightarrow \mathcal{C}_{Z/X}$ is stable under the following result in the second conditions of (1), and (3). This finishes the proof. By Definition ?? (without element is when the closed subschemes are catenary. If T is surjective we may assume that T is connected with residue fields of S . Moreover there exists a closed subspace $Z \subset X$ of X where U in X' is proper (some defining as a closed subset of the uniqueness it suffices to check the fact that the following theorem

(1) f is locally of finite type. Since $S = \mathrm{Spec}(R)$ and $Y = \mathrm{Spec}(R)$.

Proof. This is form all sheaves of sheaves on X . But given a scheme U and a surjective étale morphism $U \rightarrow X$. Let $U \cap U = \coprod_{i=1, \dots, n} U_i$ be the scheme X over S at the schemes $X_i \rightarrow X$ and $U = \lim_i X_i$. \square

The following lemma surjective restrocomposes of this implies that $\mathcal{F}_{x_0} = \mathcal{F}_{x_0} = \mathcal{F}_{X, \dots, 0}$.

Lemma 0.2. Let X be a locally Noetherian scheme over S , $E = \mathcal{F}_{X/S}$. Set $\mathcal{I} = \mathcal{I}_1 \subset \mathcal{I}_n$. Since $\mathcal{I}^n \subset \mathcal{I}^n$ are nonzero over $i_0 \leq \mathfrak{p}$ is a subset of $\mathcal{I}_{n,0} \circ \mathcal{A}_2$ works.

Lemma 0.3. In Situation ?? . Hence we may assume $\mathfrak{q}' = 0$.

Proof. We will use the property we see that \mathfrak{p} is the next functor (??). On the other hand, by Lemma ?? we see that

$$D(\mathcal{O}_{X'}) = \mathcal{O}_X(D)$$

where K is an F -algebra where δ_{n+1} is a scheme over S . \square

Примеры работы языковых моделей

```
/*
 * Increment the size file of the new incorrect UI_FILTER group information
 * of the size generatively.
 */
static int indicate_policy(void)
{
    int error;
    if (fd == MARN_EPT) {
        /*
         * The kernel blank will coeld it to userspace.
         */
        if (ss->segment < mem_total)
            unblock_graph_and_set_blocked();
        else
            ret = 1;
        goto bail;
    }
    segaddr = in_SB(in.addr);
    selector = seg / 16;
    setup_works = true;
    for (i = 0; i < blocks; i++) {
        seq = buf[i++];
        bpf = bd->bd.next + i * search;
        if (fd) {
            current = blocked;
        }
    }
    rw->name = "Getjbbregs";
    bprm_self_clearl(&iv->version);
    regs->new = blocks[(BPF_STATS << info->historidac)] | PFMR_CLOBATHINC_SECONDS << 12;
    return segtable;
}
```

The sole importance of the crossing of the Berezina lies in the fact that it plainly and indubitably proved the fallacy of all the plans for cutting off the enemy's retreat and the soundness of the only possible line of action--the one Kutuzov and the general mass of the army demanded--namely, simply to follow the enemy up. The French crowd fled at a continually increasing speed and all its energy was directed to reaching its goal. It fled like a wounded animal and it was impossible to block its path. This was shown not so much by the arrangements it made for crossing as by what took place at the bridges. When the bridges broke down, unarmed soldiers, people from Moscow and women with children who were with the French transport, all--carried on by vis inertiae--pressed forward into boats and into the ice-covered water and did not, surrender.

```

static int __dequeue_signal(struct sigpending *pending, sigset_t *mask,
    siginfo_t *info)
{
    int sig = next_signal(pending, mask);
    if (sig) {
        if (current->notifier) {
            if (sigismember(current->notifier_mask, sig)) {
                if (!(current->notifier)(current->notifier_data)) {
                    clear_thread_flag(TIF_SIGPENDING);
                    return 0;
                }
            }
        }
        collect_signal(sig, pending, info);
    }
    return sig;
}

#ifdef CONFIG_AUDITSYSCALL
static inline int audit_match_class_bits(int class, u32 *mask)
{
    int i;
    if (classes[class]) {
        for (i = 0; i < AUDIT_BITMASK_SIZE; i++)
            if (mask[i] & classes[class][i])
                return 0;
    }
    return 1;
}

```