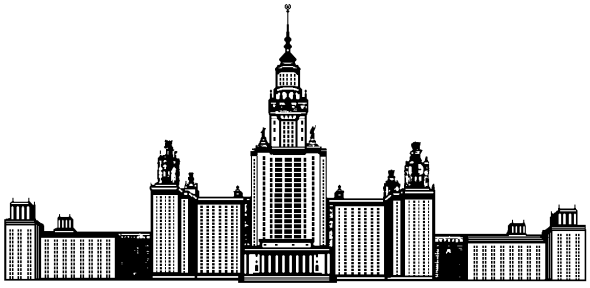

Московский государственный университет имени М.В. Ломоносова

Факультет вычислительной математики и кибернетики

Кафедра Математических Методов Прогнозирования

Рысьмятова Анастасия Александровна

# Дискриминативные и генеративные рекуррентные нейронные сети для генерации минус-слов в контекстной рекламе

МАГИСТЕРСКАЯ ДИССЕРТАЦИЯ

**Научный руководитель:**

д.ф-м.н., профессор

Дьяконов Александр Геннадьевич

Москва, 2018

# Содержание

# 1 Введение

Контекстная реклама является основным источником доходов крупнейших IT-компаний. Одним из методов улучшения качества рекламы является добавление минус-слов, запрещающих показ рекламы по запросу, содержащему выбранные слова. Минус-слова позволяют заблокировать показ баннеров по запросам, не имеющим прямого отношения к рекламируемым товарам. Например, для баннера, рекламирующего контактные линзы, необходимо добавить минус-слово «объектив», чтобы запретить показ по запросу «линзы для объектива». Минус-слова позволяют адресовать рекламу только целевой аудитории, делая её эффективнее. Именно поэтому задача генерации минус-слов так важна в этой отрасли.

Рассмотрим понятия, используемые в данной работе.

- Баннер – рекламное текстовое объявление, которое размещается на странице сайта. В данной работе рассматриваются баннеры, показываемые над поисковой выдачей. В поисковых системах для рекламного баннера определен список ключевых фраз.

- Ключевые фразы баннера – это слова или словосочетания, которые определяют, по каким поисковым запросам будет показано объявление. Будем говорить, что фраза и поисковый запрос соответствуют друг другу, если все слова фразы содержатся в поисковом запросе. На поиске баннер может показываться по запросу, если хотя бы одна ключевая фраза баннера соответствует данному запросу.

- Баннеро-фраза – это баннер и соответствующая ему ключевая фраза.

В рамках работы необходимо генерировать минус-слова для текстов баннеров, поэтому данная задача относится к задачам автоматической обработки текстов, которые в настоящее время активно решаются с использованием рекуррентных нейронных сетей [4, 13]. Гибкость рекуррентных нейронных сетей позволяет обучать их дискриминативно и генеративно [4].

В данной работе показано, что модели основанные на рекуррентных нейронных сетях могут быть использованы для решения задачи генерации минус-слов. Предложены архитектуры нейронных сетей, обучаемые дискриминативно и генеративно, позволяющие генерировать минус-слова для рекламных баннеров. Также показано, что модели на основе рекуррентных нейронных сетей справляются с задачей генерации минус-слов лучше, чем традиционный модели, основанные на TF-IDF [12] преобразовании. В рамках работы продемонстрировано, что генеративная модель превосходит дискриминативную в том случае,

если число объектов в обучающей выборке меньше, чем $10^5$. Если же в обучающей выборке более $10^5$ объектов, то с помощью дискриминативной модели достигается меньшая ошибка.

Результаты работы были доложены на XXV Международной конференции студентов, аспирантов и молодых учёных «Ломоносов-2018» [15].

Структура работы организована следующим образом:

- Во введении показана актуальность решаемой задачи. Рассмотрены основные понятия, используемые в работе. Обосновано использование методов машинного обучения для решения задачи.

- В разделе 2 представлена формальная постановка задачи. Даны определения дискриминативной и генеративной модели.

- В разделе 3 описаны основные методы генерации текстов.

- В разделе 4 объяснено понятие рекуррентной нейронной сети. Обозначены функции активации, используемые в данной работе. Описаны проблемы "взрывающегося градиента" и "затухающего градиента". А также наиболее популярные архитектуры, решающие проблему "затухающего градиента".

- В разделе 5 описан метод Word2Vec – наиболее популярный метод перевода слов в вектор фиксированной длины.

- В разделе 6 приведены результаты экспериментов, проводимых с использованием реальных данных. Показаны архитектуры нейронных сетей, решающие задачу генерации минус слов для рекламных баннеров.

- В заключении обобщены и сформулированы результаты работы.

# 2 Дискриминативные и генеративные модели

Формально задача генерации минус-слов выглядит следующим образом.

Пусть $X$ – множество баннеро-фраз, $Y$ – множество минус-слов, мощность которого равна $D$. Будем считать, что все элементы множества $Y$ пронумерованы числами от 1 до $D$. $X^\ell = \{x_1, x_2, ..., x_\ell\}$ – объекты обучающей выборки, а $\{y_1, y_2, ..., y_\ell\} : y_i \in \{0,1\}^D$ – целевые значения для объектов из множества $X^\ell$. Элемент $j$ вектора $y_i$ равен 1, если слово под номером $j$ из множества $Y$ является минус-словом для объекта $x_i$. Предположим, что обучающие объекты и ответы на них $(x_1, y_1), ..., (x_\ell, y_\ell)$ независимо выбираются из некоторого распределения $p(x, y)$, заданного на множестве $X \times \{0,1\}^D$.

Анализируя объекты из обучающей выборки $X^\ell$ и целевые значения $\{y_1, y_2, ..., y_\ell\}$, необходимо выработать решающее правило, которое позволило бы для всего множества $X$ спрогнозировать целевую переменную из множества $\{0,1\}^D$.

Традиционно в машинном обучении выделяют два вида моделей, решающих поставленную задачу:

1. Дискриминативная – моделирует условное распределение $p(y|x)$.

2. Генеративная – моделирует полное распределение $p(x, y)$.

Генеративные модели более общие: если известно $p(x, y)$, то, используя определение условной вероятности, можно найти $p(y|x)$.

$$p(y|x) = \frac{p(y, x)}{p(x)}$$

С помощью генеративных моделей возможно решать более сложные задачи, чем задача прогнозирования целевого значения. Например, можно генерировать пары $(x, y)$, чтобы создавать новые данные, аналогичные имеющимся. Или же моделировать $p(x|y)$, чтобы по целевому значению получать новые объекты.

До недавнего времени при решении задач классификации или регрессии обычно использовали дискриминативные модели. Основная причина этого, сформулированная в работе [14], заключается в том, что необходимо решать проблему (классификации или регрессии) напрямую, а не решать более общую проблему как промежуточный шаг (например, моделирование $p(x|y)$ из $p(x, y)$). Но уже в 2001 году было показано, что при использовании дискриминативной модели не во всех задачах достигается меньшая ошибка, чем при использовании генеративной модели. В статье [9] сравнивалась генеративная модель наивного байесовского классификатора и ее дискриминативный аналог – логистическая регрессии.

При увеличении числа объектов в обучении было выявлено, что зачастую генеративная модель имеет более высокую асимптотическую ошибку, чем дискриминативная, но при этом генеративная модель может сходиться к своей асимптотической ошибке намного быстрее. В настоящее время аналогичные результаты также показаны для рекуррентных нейронных сетей [4].

Успех применения генеративных моделей при решении задач классификации и регрессии, а также необходимость решать более сложные задачи автоматической обработки текстов, в которых целевой переменной может быть последовательность произвольной длины, породили множество исследований генеративных текстовых моделей.

# 3 Генеративные текстовые модели

Текст на естественном языке может быть рассмотрен как последовательность символов или же последовательность слов, в которой каждый новый элемент зависит от предыдущих. Рассмотрим задачу моделирования последовательности, а именно задачу предсказания по фиксированной части последовательности следующего элемента этой последовательности. Пусть $V$ – множество всех возможных элементов последовательности. По своему смыслу множество $V$ – это словарь. Будем считать, что элементами последовательности могут быть только элементы множества $V$.

Пусть $\widehat{x}_1, \widehat{x}_2, ..., \widehat{x}_{t-1}$ – элементы последовательности, $n$ – фиксированная часть или окно по которому необходимо предсказать новый элемент. Будем считаем, что $\widehat{x}_i$ зависит лишь от $\widehat{x}_{i-1}, \widehat{x}_{i-2}, ..., \widehat{x}_{\max(i-n,1)}$. Задача предсказания нового элемента последовательности может быть сведена к нахождению распределения (1).

$$p(x_t|\widehat{x}_{t-1}, \widehat{x}_{t-2}, ..., \widehat{x}_{\max(t-n,1)}) \tag{1}$$

Зная распределение (1) для любых значений $\widehat{x}_{t-1}, \widehat{x}_{t-2}, ..., \widehat{x}_{\max(t-n,1)}$, и, задать первый элемент последовательности (например, случайным образом) можно генерировать новую последовательность двумя способами:

1. Жадный способ. Если известны элементы последовательности $\widehat{x}_{t-1}, \widehat{x}_{t-2}, ..., \widehat{x}_{\max(t-n,1)}$, то предсказание значения нового элемента последовательности $\widehat{x}_t \in V$ вычисляется по формуле (2).

$$\widehat{x}_t = \arg\max_{x_t \in V} p(x_t|\widehat{x}_{t-1}, \widehat{x}_{t-2}, ..., \widehat{x}_{\max(t-n,1)}) \tag{2}$$

2. Лучевой поиск (Beam search). Определяются $m$ самых вероятных элементов последовательности. Генерация продолжается для всех $m$ вариантов, среди которых выбираются $m$ с максимальным значением вероятности.

Получив распределение (1), и, используя правило произведения (general product rule), можно найти совместное распределение всей последовательности определенной длины:

$$p(x_1, ..., x_k) = p(x_k|x_{k-1}, ..., x_{k-n})p(x_1, ..., x_{k-1}) = \prod_{i=1}^{k} p(x_i|x_{i-1}, ..., x_{\max(i-n,1)})$$

Сформулированная задача генерации текста может быть решена многими методами машинного обучения, но при этом в ней возникает проблема выбора ширины окна $n$. Задавая слишком маленькое окно, модель не сможет находить некоторые особенности естественного языка. Однако при больших значениях $n$ увеличивается сложность обучения модели.

В настоящее время для генерации текста на естественном языке активно применяются рекуррентные нейронные сети, которые для предсказания нового элемента последовательности используют все предыдущие элементы.

# 4 Рекуррентные нейронные сети

Глубокие нейронные сети – это алгоритм машинного обучения, который успешно применяется в различных областях, таких как классификация изображений [7] или распознавание речи [1]. Кроме того, результаты многих исследований показали, что нейронные сети также могут быть использованы при решении ряда задач обработки естественного языка.

Рекуррентные нейронные сети [5] – вид нейронных сетей, архитектура которых состоит из трех слоёв: входного, скрытого и выходного. При этом скрытый слой имеет обратную связь сам на себя. В отличие от нейронных сетей с прямой связью, рекуррентная нейронная сеть может принимать на вход последовательность векторов $x = (x_1, x_2, ..., x_T)$ в заданном порядке. Работа рекуррентной нейронной сети описывается соотношениями (3) и (4), где $\phi$ – нелинейная функция; $x_t$ – входной вектор номер $t$; $h_t$ – состояние скрытого слоя для входа $x_t$; $y_t$ – выход сети для входа $x_t$; $W, W_x, W_y$ – весовые матрицы нейронной сети; $b_h, b_y$ – векторы сдвига.

$$h_t = \phi(W h_{t-1} + W_x x_t + b_h) \tag{3}$$

$$y_t = \phi(W_y h_t + b_y) \tag{4}$$

Благодаря рекуррентному соотношению (3) имеется возможность учета результатов преобразования нейронной сетью информации на предыдущих этапах для обработки входного вектора на следующем этапе функционирования сети.

На рис. 1 представлена схема работы рекуррентной нейронной сети.

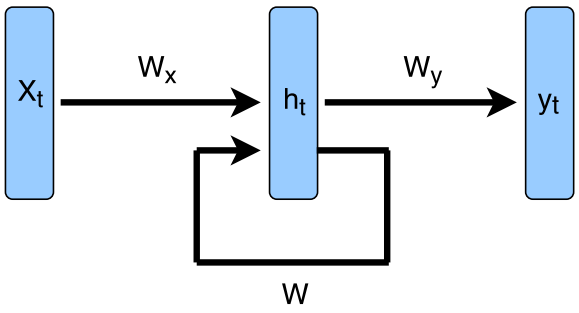


Рис. 1: Схема работы рекуррентной нейронной сети

Гибкость рекуррентных нейронных сетей позволяет обучать их дискриминативно и генеративно [4]. При генеративном обучении, применяя к выходу $y_t$ нормировочную функцию $g$, получим, что $g(y_t)$ – это распределение на следующий элемент последовательности, учитывающее знания о предыдущих элементах последовательности. Использование правила произведения (general product rule), дает возможность оценить вероятность всей последовательности:

$$g(y_t) = p(x_{t+1}|x_1, ..., x_t)$$

$$p(x_1, ..., x_T) = p(x_1)p(x_2|x_1)p(x_3|x_1, x_2)...p(x_T|x_1, ..., x_{T-1})$$

## 4.1 Функции активации

В данном разделе описаны используемые в работе функции активации для нейронных сетей.

Пусть $s$ – сигнал активации нейрона. Выделим следующие функции активации:

- $\sigma(s) = \dfrac{1}{1+e^{-s}}$ — сигмоидная
- $tanh(s) = 2\sigma(2s) - 1$ — тангенциальная
- $ReLU(s) = \max(0, s)$ — положительно линейная (ReLU)
- Софтмакс (Softmax):

  Пусть $M-$ количество нейронов в уровне с функцией активации Softmax, $s_j-$ сигнал активации на $j-$ом нейроне. Тогда выходом на $j-$ом нейроне с функцией активации Softmax будет $f_j(s_1, ..., s_M)$.

  $f_j(s_1, ..., s_M) = \dfrac{e^{s_j}}{\sum_{k=1}^{M} e^{s_k}}$ для $j = 1, ..., M$

## 4.2 Обучение рекуррентных нейронных сетей

Для обучения рекуррентных нейронных сетей [10] вводится функция потерь $\mathcal{L}_t(y_t, \widehat{y_t})$, характеризующая величину отклонения ответа $y_t$ нейронной сети от правильного ответа $\widehat{y_t}$ в момент времени $t$. Полная функция потерь для одного объекта обучающей выборки определяется по формуле (5).

$$\mathcal{L}(y, \widehat{y}) = \sum_i \mathcal{L}_i(y_i, \widehat{y_i}) \tag{5}$$

Рекуррентная нейронная сеть, обучаясь, должна минимизировать функцию потерь по всей обучающей выборке. Минимизация функции потерь обычно происходит с помощью метода стохастического градиентного спуска. Для этого необходимо вычислить $\dfrac{\partial \mathcal{L}}{\partial W}$, $\dfrac{\partial \mathcal{L}}{\partial W_x}$, $\dfrac{\partial \mathcal{L}}{\partial W_y}$ с помощью алгоритма обратного распространения ошибки сквозь время [10] (Backpropagation Through Time). В данном алгоритме разворачивается граф вычислений во времени, как на рис. 2.

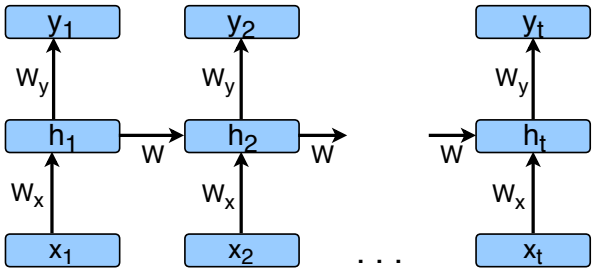


Рис. 2: Схема работы рекуррентной нейронной сети

Из рис. 2 можно заметить, что верны равенства (6), (7), (8).

$$\frac{\partial \mathcal{L}}{\partial W} = \sum_t \frac{\partial \mathcal{L}_t(y_t, \widehat{y_t})}{\partial W} \tag{6}$$

$$\frac{\partial \mathcal{L}_t}{\partial W} = \frac{\partial \mathcal{L}_t}{\partial y_t} \frac{\partial y_t}{\partial h_t} \frac{\partial h_t}{\partial W} \tag{7}$$

$$\frac{\partial h_t}{\partial W} = \sum_{k=0}^{t} \left( \prod_{i=k+1}^{t} \frac{\partial h_i}{\partial h_{i-1}} \right) \frac{\partial h_k}{\partial W} \tag{8}$$

Если $\forall i$ выполнено $\left\| \frac{\partial h_i}{\partial h_{i-1}} \right\|_2 > 1$, возникает проблема "взрывающегося градиента" [10] (exploding gradients). Для борьбы с этой проблемой обычно используют обрезание градиента, когда его норма превышает определенный порог (gradient clipping).

Если $\forall i$ выполнено $\left\| \frac{\partial h_i}{\partial h_{i-1}} \right\|_2 < 1$, то возникает проблема "затухания градиента" [10] (vanishing gradients). Чем дальше элемент находится от текущего, тем меньше его вклад в градиент. В этом случае $\left\| \prod_{i=k+1}^{t} \frac{\partial h_i}{\partial h_{i-1}} \right\|_2$ будет экспоненциально стремиться к нулю с ростом $t-k-1$. Одним из методов борьбы с затухающим градиентом является изменение формулы вычисления скрытого состояния $h_t$ (3). Наиболее популярными архитектурами рекуррентных нейронных сетей, помогающими избежать проблему затухающего градиента, являются LSTM и GRU сети.

## 4.3 LSTM

LSTM (Long Short-Term Memory) или сеть с долговременно-кратковременной памятью [5] – особый тип рекуррентных нейронных сетей. В отличие от блока простой рекуррентной сети, который вычисляет взвешенную сумму входного сигнала и применяет нелинейную функцию, каждый блок LSTM сети имеет память $c_t$, которая хранит информацию, полученную в предыдущий момент времени. В LSTM сети скрытый слой $h_t$ вычисляется по

формуле (9), $o_t$ – выходной шлюз (output gate), который контролирует объем данных, получаемых из памяти, и вычисляется по формуле (10). В формулах символом $\odot$ обозначается поэлементное произведение векторов.

$$h_t = o_t \odot \tanh(c_t) \tag{9}$$

$$o_t = \sigma(W_o x_t + U_o h_{t-1} + v_o \odot c_t + b_o) \tag{10}$$

Память $c_t$ обновляется по формуле (11), где $f_t$ – шлюз забывания (forget gate), который контролирует степень сохранения данных из памяти прошлого блока, а $i_t$ – входной шлюз (input gate), определяющий влияние промежуточной памяти $\widehat{c_t}$ на результат.

$$c_t = f_t \odot c_{t-1} + i_t \odot \widehat{c_t} \tag{11}$$

$$\widehat{c_t} = \tanh(W_c x_t + U_c h_{t-1} + b_c) \tag{12}$$

Значения $f_t$ и $i_t$ вычисляются по формулам (13) и (14) соответственно.

$$f_t = \sigma(W_f x_t + U_f h_{t-1} + v_f \odot c_t + b_f) \tag{13}$$

$$i_t = \sigma(W_i x_t + U_i h_{t-1} + v_i \odot c_t + b_i) \tag{14}$$

В перечисленных формулах $x_t$ — входной вектор текущего блока LSTM сети; $h_{t-1}$ — выход предыдущего блока LSTM сети; $W_f, W_c, W_i, W_o, U_f, U_c, U_i, U_o$ — матрицы, настраиваемые нейронной сетью; $b_f, b_c, b_i, b_o, v_f, v_i, v_o$ – векторы, настраиваемые нейронной сетью.

## 4.4 GRU

GRU (Gated Recurrent Unit) или сеть с циклически повторяющимся блоком [3]. Подобно LSTM, GRU состоит из блоков, которые моделируют информацию внутри себя, однако, без наличия отдельных ячеек памяти. GRU использует меньше операций для вычисления, чем LSTM. В GRU сети скрытый слой $h_t$ вычисляется по формуле (15), на основе промежуточного значения $\widehat{h_t}$ (16) и шлюза обновления $z_t$ (update gate) (17).

$$h_t = z_t \odot h_{t-1} + (1 - z_t) \odot \widehat{h_t} \tag{15}$$

$$\widehat{h_t} = tanh(W x_t + U(r_t \odot h_{t-1})) \tag{16}$$

$$z_t = \sigma(W_z x_t + U_z h_{t-1}) \tag{17}$$

Шлюз сброса состояния $r_t$ (reset gate) контролирует степень сохранения данных из прошлого блока в промежуточном значении $\widehat{h_t}$ и вычисляется по формуле (18).

$$r_t = \sigma(W_r x_t + U_r h_{t-1}) \tag{18}$$

# 5 Векторное представление слов

Рекуррентная нейронная сеть принимает на вход последовательность векторов $x = (x_1, x_2, ..., x_T)$, поэтому для использования рекуррентных нейронных сетей в задачах автоматической обработки текстов необходимо закодировать каждое слово либо каждый символ текста с помощью вектора фиксированной длины. Наиболее распространенным способом сопоставления слова вектору является нейросетевая технология Word2Vec.

## 5.1 Word2Vec

Word2Vec [2] — технология от компании Google, которая предназначена для статистической обработки больших массивов текстовой информации. Word2Vec собирает статистику о появлении слов в данных, удаляет наиболее редко и часто встречаемые слова, после чего с помощью нейронных сетей решает задачу снижения размерности и выдает на выходе компактные векторные представления слов заранее определенной длины.

Пусть $V$ – количество различных слов в тексте (объем словаря), $T$ – общее число слов в данных, $N$ – необходимая длина векторного представления слова. Каждое слово из словаря кодируется бинарным вектором размера $V$ с одной единицей. Обозначим через $w_t$ слово текста на позиции $t$. Контекстом $w_t$ назовем все слова $w_i : i \in [\max(1, t-C/2), t) \cup (t, min(t+C/2, T)]$, где $C$ – гиперпараметр модели.

В Word2Vec возможно использование двух различных архитектур нейронной сети, предназначенных для перевода слова в вектор: Continuous Bag of Words (CBOW) и Skipgram.

### 5.1.1 Continuous Bag of Words

При использовании архитектуры Continuous Bag of Words нейронная сеть предсказывает слово $w_t$ по его контексту. Обучаемая нейронная сеть состоит из трех слоёв: на входном (input) слое – $C \cdot V$ нейронов, на выходном (output) слое – $V$ нейронов, на скрытом (hidden) слое – $N$ нейронов. Связи между слоями сети полносвязные. Функция активации на выходном слое – софтмакс (softmax). На рис. 3 приведена схема архитектуры сети.

Веса между скрытым и выходным слоем можно представить в виде матрицы $W'_{N \times V}$. После обучения нейронной сети столбцы матрицы $W'_{N \times V}$ – вектора длины $N$ будут результатом работы алгоритма CBOW технологии Word2Vec. Нейронная сеть, обучаясь, максимизирует функцию:

$$\prod_{i=1}^{T} p(w_i | w_{\max(i-C/2,1)}, ..., w_{i-1}, w_{i+1}, ..., w_{\min(i+C/2,T)})$$

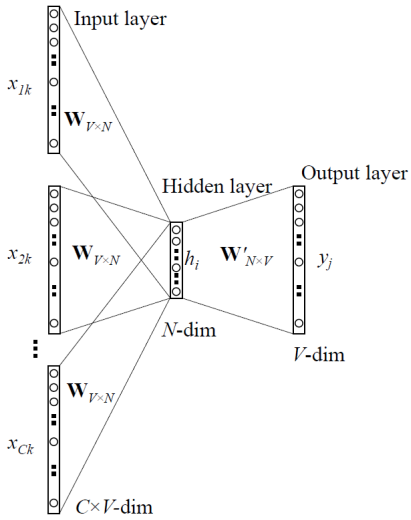


Рис. 3: Архитектура сети для CBOW модели [11]

### 5.1.2 Skipgram

При использовании архитектуры Skipgram нейронная сеть по слову $w_t$ предсказывает контекст этого слова. На рис. 4 приведен пример архитектуры сети для Skipgram модели. Веса между скрытым и входным слоем можно представить в виде матрицы $W'_{V\times N}$. После

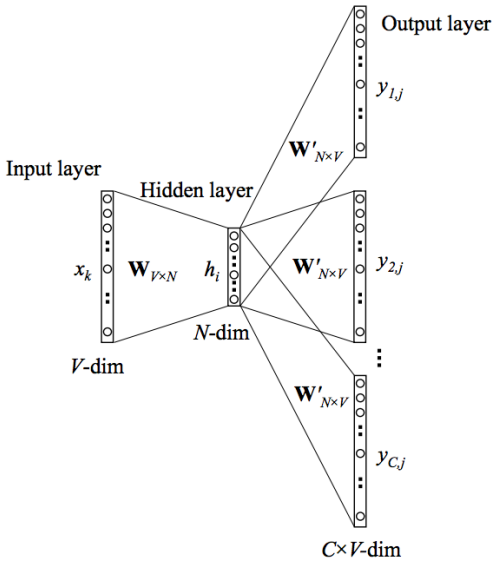


Рис. 4: Архитектуры сети для skipgram модели [11]

обучения нейронной сети строки матрицы $W'_{V\times N}$ – вектора длины $N$ будут результатом работы алгоритма Skipgram технологии Word2Vec.

Нейронная сеть, обучаясь, максимизирует функцию:

$$\prod_{i=C/2}^{T} \prod_{-\frac{C}{2}<j<\frac{C}{2}, j\neq 0} p(w_{i+j}|w_i)$$

# 6 Эксперименты

В данной работе была исследована задача генерации минус-слов для рекламных баннеров с помощью рекуррентных нейронных сетей.

## 6.1 Данные

Для проведения экспериментов была собрана статистика кликов по рекламным баннерам, которые показывались над поисковой выдачей для запросов за последние полгода.

Пусть $B = \{b_1, b_2, ..., b_M\}$ – множество рекламных баннеров, для каждого баннера $b \in B$ определено $F(b) = \{f_1, f_2, ..., f_d\}$ – множество ключевых фраз баннера $b$. Для каждого рекламного баннера $b \in B$ и каждой фразы $f \in F(b)$ из собранной статистики было найдено множество слов $W(b, f)$, содержащихся в поисковых запросах, по которым был показан баннер $b$, но не содержащихся в ключевой фразе $f$. Для всех слов $w \in W(b, f)$ вычислено:

- shows – число показов баннера $b$ по поисковым запросам, соответствующим фразе $f$ и содержащим слово $w$;
- clicks – число кликов на баннер $b$ по поисковым запросам, соответствующим фразе $f$ и содержащим слово $w$;
- метрика CTR, которая определяется как отношение $\dfrac{\text{clicks}}{\text{shows}}$ и измеряется в процентах;

Выявлены баннеро-фразы и соответствующие им слова $w \in W(b, f)$ с числом показов shows $> 100$. Из полученного набора выделены данные для которых CTR $< 2\%$ и CTR $> 15\%$. Ограничение на число показов и пороги CTR были определены эмпирически. Таким образом, была получена выборка, содержащая $2 \cdot 10^6$ объектов. Каждый объект включает в себя три показателя ($b$, $f$, $w$). Целевое значение объекта $y$ определялось по формуле (19).

$$y = \begin{cases} 1, & \text{если CTR слова } w < 2\% \\ 0, & \text{если CTR слова } w > 15\% \end{cases} \tag{19}$$

Далее выборка была разделена случайным образом на обучающую и контрольную. Для проверки качества работы было использовано 50000 объектов. Помимо использования отложенной выборки, качество обученной модели проверялось на данных, размеченных асессорами.

## 6.2 Предобработка данных

Для решения поставленной задачи использовались слова текстового описания баннеров и ключевых фраз. Перед использованием моделей машинного обучения все слова были приведены в начальную форму, а затем каждое слово было закодировано с помощью вектора фиксированной длины. Для этого случайным образом было выбрано $5 \cdot 10^9$ баннеров и обучена Skipgram Word2Vec модель на текстах баннеров и их ключевых фразах.

## 6.3 Модели

В рамках работы построены дискриминативная и генеративная модели на основе рекуррентных нейронных сетей.

Для каждого объекта $(b,\ f,\ w)$ обозначим $T = \{t_1, ..., t_m\}$ – векторные представления слов текстового описание баннера $b$, $P = \{p_1, ..., p_\ell\}$ – векторные представления слов ключевой фразы $f$, $\overline{w}$ – вектор слова $w \in W(b, f)$, $y$ – целевое значение.

$$y = \begin{cases} 1, & \text{если } w \text{ минус-слово для баннеро-фразы } (b, f) \\ 0, & \text{если } w \text{ не минус-слово для баннеро-фразы } (b, f) \end{cases}$$

Для заданной обучающей выборки $\{T^{(i)}, P^{(i)}, w^{(i)}, y^{(i)}\}_{i=1}^N$ построенная дискриминативная модель, обучаясь, максимизирует сумму логарифмов условного распределения по всей обучающей выборке (20).

$$\sum_{i=1}^{N} \log p(y^{(i)} | w^{(i)}, T^{(i)}, P^{(i)}) \tag{20}$$

Генеративная модель во время обучения максимизирует сумму логарифмов совместного распределения по всей обучающей выборке (21).

$$\sum_{i=1}^{N} \log p(y^{(i)}, w^{(i)}, T^{(i)}, P^{(i)}) = \sum_{i=1}^{N} \log p(P^{(i)} | y^{(i)}, w^{(i)}, T^{(i)}) p(T^{(i)} | y^{(i)}, w^{(i)}) p(w^{(i)} | y^{(i)}) p(y^{(i)}) \tag{21}$$

### 6.3.1 Дискриминативная модель

Архитектура дискриминативной модели следующая. Векторы текстового описание баннера передаются в рекуррентную нейронную сеть GRU.

$$\{\widehat{t}_1, \widehat{t}_2, ..., \widehat{t}_m\} = GRU\{t_1, ..., t_m\}$$

Векторы фразы передаются в другую рекуррентную нейронную сеть GRU.

$$\{\widehat{p}_1, \widehat{p}_2, ..., \widehat{p}_\ell\} = GRU\{p_1, ..., p_\ell\}$$

Выходы последних скрытых слоев объединяются и подаются на вход полносвязному слою с функцией активации $ReLU$. Количество нейронов на выходе полносвязного слоя $\overline{u}$ равно размерности векторного представления слов.

$$\overline{u} = ReLU(Fully\_connected\{[\widehat{t}_m, \widehat{p}_\ell]\})$$

Вектор полносвязного слоя $\overline{u}$ и вектор $\overline{w}$ объединяются и подаются на вход полносвязному слою с функцией активации Softmax. На выходе полносвязного слоя два нейрона.

$$p(y|T, P, w, \theta_1, \theta_2) = Softmax(Fully\_connected\{[\overline{u}, \overline{w}]\})$$

Здесь $\theta_1, \theta_2$ – настраиваемые параметры нейронной сети, $[,]$ – операция объединения векторов, Fully_connected{} – полносвязный слой. Полученное распределение $p(y|T, P, w, \theta_1, \theta_2)$ передается в функцию потерь. Схема работы описанной модели представлена на рис. 5.

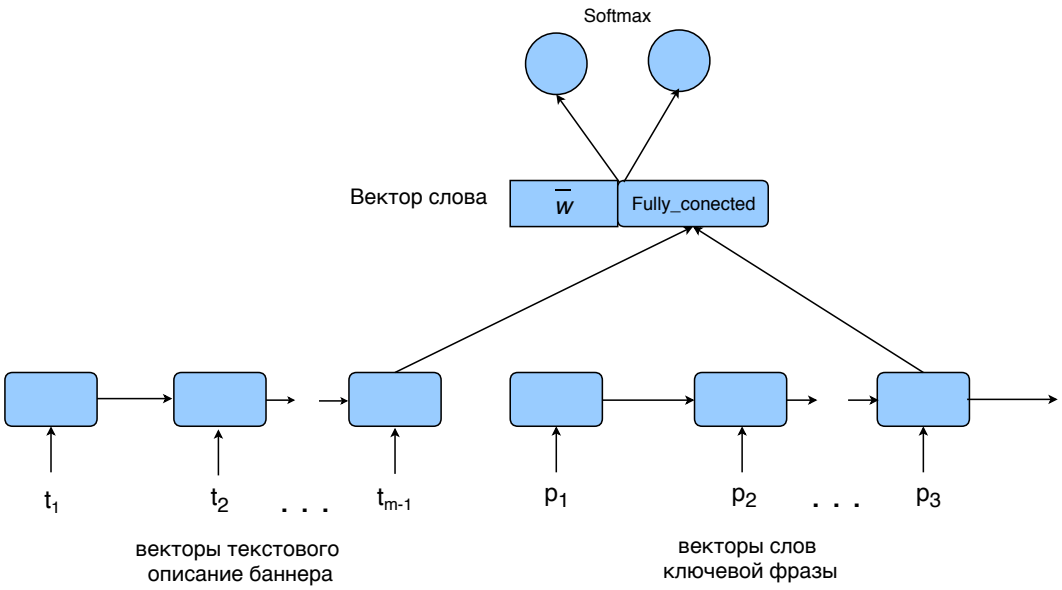


Рис. 5: Архитектура дискриминативной нейронной сети

### 6.3.2 Генеративная модель

Генеративная модель, обучаясь, максимизирует совместное распределение $p(y, T, P, w, \theta)$, где $\theta$ параметры обучаемой модели. Используя правило произведения вероятностей, получаем равенство (22).

$$p(y, T, P, w, \theta) = p(P|y, w, T, \theta)p(T|y, w, \theta)p(w|y)p(y)$$

$$= \prod_{i=1}^{m} p(p_i|p_{<i}, y, w, T, \theta) \prod_{i=1}^{m} p(t_i|t_{<i}, y, w, \theta) p(w|y) p(y) \tag{22}$$

Предположим, что текстовое описание баннера $T$ не зависит от слова $w$ и целевой переменной $y$. Это позволяет уменьшить время обучения нейронной сети, не увеличивая ошибку. Тогда справедливо равенство (23).

$$p(y, T, P, w, \theta_1, \theta) = \prod_{i=1}^{m} p(p_i|p_{<i}, y, w, P, \theta) p(T|\theta) p(w|y) p(y) \tag{23}$$

Для предсказания целевой переменной $y$ нового объекта необходимо вычислить (24).

$$y = \arg\max_{y \in \{0,1\}} p(y|T, P, w, \theta) = \arg\max_{y \in \{0,1\}} \frac{p(y, T, P, w, \theta)}{p(T, P, w, \theta)} = \arg\max_{y \in \{0,1\}} p(y, T, P, w, \theta)$$

$$= \arg\max_{y \in \{0,1\}} p(P|y, w, T, \theta) p(T|\theta) p(w|y) p(y) = \arg\max_{y \in \{0,1\}} p(P|y, w, T, \theta) p(w|y) p(y) \tag{24}$$

При построении генеративной модели за основу была взята архитектура нейронной сети из статьи [4]. Генеративная модель состоит из двух рекуррентных нейронных сетей GRU. Первая сеть принимает на вход все слова текстового описания баннера и сопоставляет тексту баннера вектор фиксированной длины. Вторая рекуррентная сеть на каждой итерации получает векторы слов ключевой фразы и пытается предсказывать следующее слово фразы баннера. Построенная модель представлена на рис. 6.

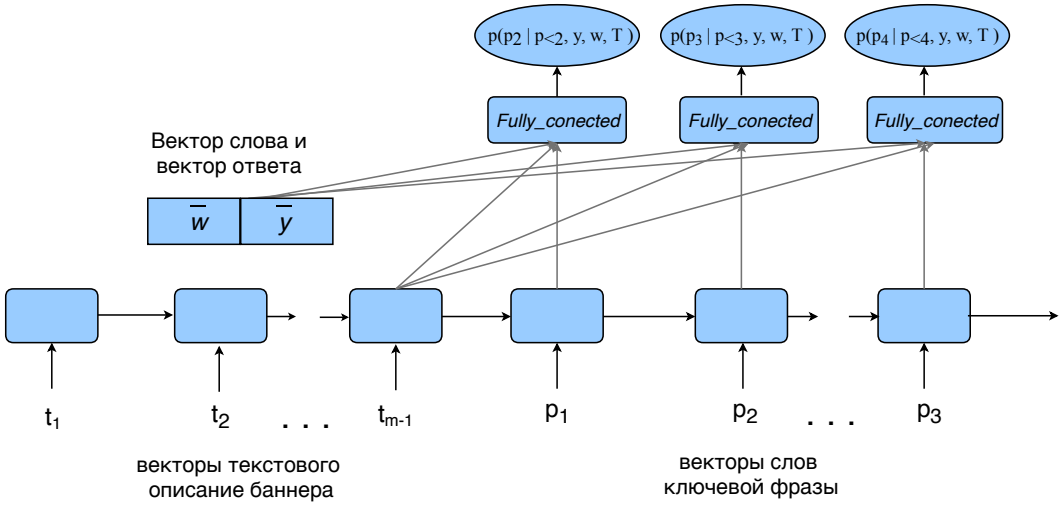


Рис. 6: Архитектура генеративной нейронной сети

Опишем формально генеративную модель. Первая рекуррентная сеть принимает на вход векторы $t_1, ..., t_m$. Вектор последнего скрытого слоя $\widehat{t}_m$ передается второй рекуррентной сети, как начальный вектор скрытого состояния.

$$\{\widehat{t}_1, \widehat{t}_2, ..., \widehat{t}_m\} = GRU\{t_1, ..., t_m\}$$

Вторая рекуррентная сеть принимает на вход векторы $p_1, ..., p_\ell$.

$$\{\widehat{p}_1, \widehat{p}_2, ..., \widehat{p}_\ell\} = GRU\{p_1, ..., p_\ell\}$$

Вектор скрытого состояния рекуррентной сети на каждом такте объединяется с вектором слова $\overline{w}$, вектором ответа $\overline{y}$ и вектором текста баннера $\widehat{t}_m$. Полученный вектор передается на вход полносвязному слою с функцией активации Softmax, который выдает распределение на следующее слово ключевой фразы баннера (25). В формуле (25) $\theta_2$ – это настраиваемые параметры нейронной сети.

$$p(p_i|p_{<i}, y, w, T, \theta_2) = Softmax(Fully\_connected\{[\widehat{p}_{i-1}, \overline{w}, \overline{y}]\}) \tag{25}$$

Полученные распределения $p(p_i|p_{<i}, y, w, T, \theta_2)$, где $i = \overline{1, \ell - 1}$ передаются в функцию потерь. Целевая переменная для объекта из отложенной выборки определяется по формуле.

$$y = \arg\max_{y \in \{0,1\}} \prod_{i=1}^{m} p(p_i|p_{<i}, y, w, T, \theta) p(w|y) p(y)$$

Вместо $p(w|y)$ и $p(y)$ использовались оценки эмпирической частоты, вычисленные по обучающей выборке.

## 6.4 Механизм внимания

В описанных архитектурах в рекуррентных сетях использовались лишь векторы последнего скрытого слоя. Во многих работах показано, что при использовании всех векторов скрытого слоя рекуррентной нейронной сети $\widehat{t}_{align}$ (26) можно добиться меньшей ошибки [8].

$$\widehat{t}_{align} = \sum_{j=1}^{m} \alpha_j \widehat{t}_j \tag{26}$$

В данной работе (аналогично работе [8]) использовалась поэлементная сумма всех векторов скрытых слоёв $\widehat{t}_i$ с коэффициентами внимания $\alpha_i$ (attention) (27).

$$\alpha_i = \frac{\exp(s_i)}{\sum_{j=1}^{m} \exp(s_j)} \tag{27}$$

Каждый коэффициент внимания $\alpha_i$ (27) представляет собой нормированную линейную комбинацию значений вектора скрытого слоя в момент времени $i$ и векторного представления слова $\overline{w}$ с обучаемыми весами (28).

$$s_i = Fully\_connected\{[\widehat{t}_i, \overline{w}]\} \tag{28}$$

## 6.5 Результаты

Во всех реализованных моделях размер вектора скрытого представления использован равным 100. Обучение проводилось с помощью метода оптимизации ADAM [6] с шагом обучения $10^{-4}$. Описанные модели были реализованы на языке Python с использованием библиотеки Pytorch.

Для сравнения качества работы рекуррентных нейронных сетей была реализована базовая модель, основанная на TF-IDF [12] преобразовании текстового описания баннеров, ключевых фраз и слов. На полученных TF-IDF векторах обучалась линейная модель.

Для сравнения качества моделей использовались следующие метрики качества: доля правильных ответов (Accuracy), точность (Precision), полнота (Recall), $F_1$ мера ($F_1$ score).

Качество работы моделей на отложенной выборке приведены в Таблице 1.

Таблица 1: Качество работы моделей на отложенной выборке.

| | Accuracy | Precision | Recall | $F_1$ мера |
|---|---|---|---|---|
| Дискриминативная модель | 0.79 | 0.85 | 0.72 | 0.78 |
| Дискриминативная модель с вниманием | 0.80 | 0.86 | 0.74 | 0.80 |
| Генеративная модель | 0.77 | 0.80 | 0.70 | 0.75 |
| TF-IDF модель | 0.71 | 0.70 | 0.68 | 0.69 |

Для определения качества работы построенных моделей использовались асессорские оценки – оценки качества выдачи рекламы по поисковому запросу, которые выставляют люди, следуя специальным инструкциям. Асессор анализирует насколько баннер релевантен поисковому запросу и ставит одну из меток PERFECT, GOOD, BAD, HORRIBLE . Чтобы применить обученные модели к данным с асессорскими оценками, в каждом поисковом запросе были найдены слова, которые не содержались в ключевой фразе баннера, соответствующей данному поисковому запросу. Построенные модели определяют является ли хоть одно из найденных слов минус-словом. Если модель находит минус-слово в поисковом запросе, то данный объект удаляется. Необходимо, чтобы среди удаленных объектов было как можно больше объектов с оценкой HORRIBLE и как можно меньше объектов с оценкой PERFECT.

В Таблице 2 представлено распределение оценок в используемых данных, размеченных асессорами в процентном соотношении. В Таблице 3 представлена доля удаленных объектов для каждой оценки в процентном соотношении.

Из приведенных результатов видно, что построенная дискриминативная модель на ос-

Таблица 2: Распределение асессорских оценок в процентном соотношении.

| PERFECT | GOOD | BAD | HORRIBLE |
|---|---|---|---|
| 26% | 48% | 18% | 8% |

Таблица 3: Доля удаленных объектов в процентном соотношении.

| | PERFECT | GOOD | BAD | HORRIBLE |
|---|---|---|---|---|
| Дискриминативная модель | 2% | 5% | 9% | 17% |
| Дискриминативная модель с вниманием | 2% | 5% | 10% | 17% |
| Генеративная модель | 3% | 5% | 8% | 12% |
| TF-IDF модель | 5% | 8% | 9 % | 10 % |

нове рекуррентных сетей позволяет достичь меньшей ошибки на отложенной выборке. Результат работы моделей на асессорских оценках подтверждает, что модели на основе рекуррентых нейронных сетей справляются с задачей. С точки зрения времени, обучения дискриминативная модель работает значительно быстрее генеративной. Это происходит из-за вычисления Softmax слоя в генеративной рекуррентной сети, который выдает распределение на следующее слово текста.

На рис. 7 приведены графики зависимости доли правильных ответов (Accuracy) на отложенной выборке для дискриминативной и генеративной модели в зависимости от числа объектов в обучении.

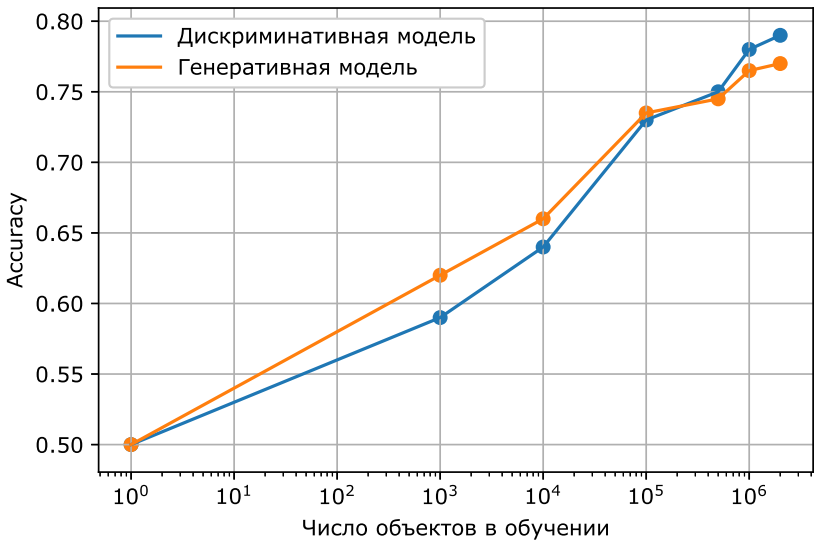


Рис. 7: Зависимость Accuracy на отложенной выборке для дискриминативной и генеративной модели в зависимости от числа объектов в обучении

Из рис. 7 видно, что генеративная модель превосходит дискриминативную, когда объектов в обучающей выборке меньше $10^5$, но когда в обучающей выборке более $10^5$ объектов, то дискриминативная модель достигает меньшей ошибки. Из проведенного исследования можно сделать вывод, что при использовании генеративных рекуррентных нейронных сетей для решения задач автоматической обработки текстов на небольших выборках можно достичь меньшей ошибки, чем при использовании дискриминативных.

## 6.6 Выводы

На основании результатов проведенных экспериментов можно сделать следующие выводы:

- Рекуррентные нейронные сети являются гибким алгоритмом машинного обучения, который можно обучать дискриминативно и генеративно.
- Рекуррентные нейронные сети в задачах автоматической обработки текстов способны достигать меньшей ошибки, чем традиционные модели основанные на TF-IDF [12] преобразовании.
- В исследуемой задаче при использовании дискриминативной модели удалось достичь меньшей ошибки, чем при использовании генеративной.
- Показано, что при небольших наборах данных (до нескольких сотен тысяч) лучше работают генеративные модели. Когда данных становится больше (более нескольких сотен тысяч объектов), лучше работают дискриминативные модели на основе рекуррентных нейронных сетей.
- При использовании механизма внимания удается достичь меньшей ошибки.

# 7 Заключение

В настоящий момент генерация минус-слов для баннеро-фраз контекстной рекламы в крупных поисковых системах осуществляется без использования методов машинного обучения. В работе было предложено решение задачи с использованием методов машинного обучения.

Также в данной работе:

1. Исследовано применение рекуррентных нейронных сетей для задач автоматической обработки текстов.

2. Доказана эффективность рекуррентных нейронных сетей в решении задач генерации минус-слов, а так же предложены архитектуры нейронных сетей, обучаемые дискриминативно и генеративно, позволяющие генерировать минус-слова для рекламных баннеров.

3. Обосновано, что модели на основе рекуррентных нейронных сетей справляются с задачей генерации минус-слов лучше, чем традиционные модели, основанные на TF-IDF [12] преобразовании.

4. Выявлены границы эффективности применения генеративной и дискриминативной модели нейронных сетей. Доказано, что генеративная модель превосходит дискриминативную, если число объектов в обучающей выборке меньше, чем $10^5$. Если же в обучающей выборке более $10^5$ объектов, то с помощью дискриминативной модели достигается меньшая ошибка.

5. Исследовано влияние использования механизма внимания на качество работы дискриминативной модели. Показано, что использование механизма внимания позволяет уменьшить ошибку модели.

# Список литературы


[1] Context-dependent pre-trained deep neural networks for large-vocabulary speech recognition. / George E. Dahl, Dong Yu, Li Deng, Alex Acero // IEEE Trans. Audio, Speech & Language Processing. — 2012. — Vol. 20, no. 1. — P. 30–42. — http://dblp.uni-trier.de/db/journals/taslp/taslp20.html#DahlYDA12.

[2] Efficient estimation of word representations in vector space / Tomas Mikolov, Kai Chen, Greg Corrado, Jeffrey Dean // ICLR. — 2013.

[3] Empirical evaluation of gated recurrent neural networks on sequence modeling. / Junyoung Chung, Caglar Gulcehre, KyungHyun Cho, Yoshua Bengio // cite arxiv:1412.3555Comment: Presented in NIPS 2014 Deep Learning and Representation Learning Workshop. — 2014. — Vol. abs/1703.01898.

[4] Generative and discriminative text classification with recurrent neural networks. / Dani Yogatama, Chris Dyer, Wang Ling, Phil Blunsom // CoRR. — 2017. — Vol. abs/1703.01898.

[5] Hochreiter, S. Long short-term memory / Sepp Hochreiter, Jurgen Schmidhuber // Neural computation. — 1997. — Vol. 9, no. 8. — P. 1735–1780.

[6] Kingma, D. P. Adam: A method for stochastic optimization. / Diederik P. Kingma, Jimmy Ba // CoRR. — 2014. — Vol. abs/1412.6980. — http://dblp.uni-trier.de/db/journals/corr/corr1412.html#KingmaB14.

[7] Krizhevsky, A. Imagenet classification with deep convolutional neural networks / Alex Krizhevsky, Ilya Sutskever, Geoffrey E. Hinton // NIPS. — 2012. — 1106 -1114 p.

[8] Learning recurrent span representations for extractive question answering / Kenton Lee, Tom Kwiatkowski, Ankur P. Parikh, Dipanjan Das // CoRR. — 2016. — Vol. abs/1611.01436. — http://arxiv.org/abs/1611.01436.

[9] Ng, A. Y. On discriminative vs. generative classifiers: A comparison of logistic regression and naive bayes / Andrew Y. Ng, Michael I. Jordan // Advances in Neural Information Processing Systems 14 (NIPS 2001) / Ed. by Thomas G. Dietterich, Suzanna Becker, Zoubin Ghahramani. — MIT Press, 2001. — P. 841–848. — http://robotics.stanford.edu/ ang/papers/nips01-discriminativegenerative.pdf.

[10] Pascanu, R. On the difficulty of training recurrent neural networks / Razvan Pascanu, Tomas Mikolov, Yoshua Bengio // International Conference on Machine Learning. — 2013. — P. 1310–1318. — http://www.jmlr.org/proceedings/papers/v28/pascanu13.pdf.

[11] Rong, X. word2vec parameter learning explained / Xin Rong // arXiv:1411.2738. — 2014.

[12] S, J. K. A statistical interpretation of term specificity and its application in retrieval / Jones K. S // Journal of Documentation. — 1972.

[13] Sutskever, I. Sequence to sequence learning with neural networks / Ilya Sutskever, Oriol Vinyals, Quoc V Le // Advances in neural information processing systems. — 2014. — P. 3104–3112. — https://papers.nips.cc/paper/5346-sequence-to-sequence-learning-with-neural-networks.pdf.

[14] Vapnik, V. N. Statistical learning theory / Vladimir N Vapnik. Adaptive and learning systems for signal processing, communications and control series. — John Wiley & Sons, New York. A Wiley-Interscience Publication, 1998.

[15] Рысьмятова, А. Дискриминативные и генеративные рекуррентные нейронные сети для генерации минус-слов в контекстной рекламе / Анастасия Рысьмятова // XXV Международная конференция студентов, аспирантов и молодых учёных «Ломоносов-2018». — 2018.