Московский государственный университет имени М.В.Ломоносова
Факультет вычислительной математики и кибернетики
Кафедра Системного Программирования

Строев Вячеслав Игоревич

Метод ранжирования ответов в вопросно-ответной системе

Выпускная квалификационная работа

Научный руководитель:
канд. физ.-мат. наук
Турдаков Денис Юрьевич

Москва, 2018

Аннотация

Метод ранжирования ответов в вопросно-ответной системе

Строев Вячеслав Игоревич

В данной работе ставится и решается задача ранжирования ответов для произвольного вопроса согласно их релевантности. Рассматриваются существующие методы решения данной и схожей задачи на основе машинного обучения и разрабатывается собственный метод, использующий нейронную сеть. Проводится оценка качества разработанного метода.

Содержание

1 Введение **5**

1.1 Семинар по оценке вопросно-ответных систем SemEval 7

1.2 Методы оценки вопросно-ответных систем 7

1.2.1 Mean average precision 7

1.2.2 Mean reciprocal rank 8

1.2.3 Average recall 9

1.3 Данные 9

2 Постановка задачи **11**

3 Обзор существующих решений **12**

3.1 Модели векторного представления слов 12

3.2 Рекуррентные нейронные сети 13

3.3 Модель KeLP 14

3.3.1 Способ ранжирования 14

3.3.2 Ядра 14

3.3.3 Парные сходства 15

3.3.4 Специфичные признаки 15

3.4 Модель Beihang-MSRA 16

3.4.1 Способ ранжирования 16

3.4.2 Подготовка 16

3.4.3 Традиционные признаки 16

3.4.4 Специфичные признаки 17

3.4.5 Нейронная сеть 18

3.5 Модель FuRongWang 19

3.5.1 Способ ранжирования 19

3.5.2 Архитектура нейронной сети 19

3.5.3 Interaction Layer 20

3.5.4 Признаки 21

3.6 Выводы 21

4 Исследование и построение решения задачи **22**
4.1 Оценка качества 22
4.2 Архитектура нейронной сети 22
4.3 Обучение нейронной сети 23
4.4 Признаки 23
4.4.1 Признаки слов 23
4.4.2 Общие признаки 24
4.5 Модификации архитектуры 24
4.6 Эксперименты 28
4.6.1 Сравнение векторных представлений 28
4.6.2 Подбор оптимальной архитектуры 29
4.6.3 Дополнительные признаки 30
4.6.4 Сравнение с существующими решениями 31
4.7 Выводы 32

5 Описание практической части **33**
5.1 Обоснование выбранного инструментария 33
5.2 Общая схема работы 33
5.3 Общая архитектура системы 35

Заключение **37**

Список литературы **38**

1 Введение

В последнее время в обществе все больше нарастает интерес к диалоговым системам. Это обусловлено тем, что диалог – естественный способ общения между людьми и, таким образом, диалог является наиболее удобным способом общения человека с машиной. К тому же, последние достижения в области обработки естественных языков, а также в области машинного обучения позволяют создавать все более универсальные и эффективные диалоговые системы.

Среди диалоговых систем можно выделить вопросно-ответные системы. Они отличаются тем, что в их задачи не входит поддержание полноценного диалога, вместо этого эти системы должны отвечать на заданные им вопросы. Таким образом, вопросно-ответные системы могут быть проще в реализации и при этом они уже могут выполнять полезную работу.

Вопросно ответные системы могут быть использованы для таких задач как:

- Предоставление экспертной консультации в различных предметных областях, таких как юридическая или медицинская помощь.
- Предоставление технической поддержки пользователю.
- Автоматизированное предоставление пользователю доступа к различным знаниям.

Сейчас в сфере обработки естественных языков достигнуты большие результаты во многих задачах, от задач уровня морфологии [1] и синтаксиса [2] до семантического уровня [3]. Таким образом, можно использовать современные методы обработки естественных языков для работы с вопросно-ответными системами.

Вопросно-ответные системы можно условно разделить на те, которые сами генерируют текст ответа и те, которые выбирают ответ из некоторой базы ответов. Первый тип позволяет системе более гибко отвечать на вопросы, и это больше похоже на естественный диалог для пользователя. Однако второй тип гораздо проще в реализации.

Основной проблемой первого типа вопросно-ответных систем является то, что сама по себе генерация осмысленного и связного текста, который будет удовлетворять контексту и правильно отвечать на поставленный вопрос – довольно сложная задача.

На данный момент неизвестно решение, которое выполняет эту задачу. Поэтому далее рассматриваются системы второго типа.

В таком случае появляется проблема выбора релевантного вопросу ответа из общего множества ответов. Эта задача может решаться многими способами, такими как подход на основе правил или на машинном обучении. Преимуществом систем на основе машинного обучения является более простая адаптация к другим предметным областям и языкам. В связи с этим в данной работе будут рассматриваться именно те подходы, которые используют машинное обучение.

В вопросно ответных системах второго типа предполагается, что будет проведено ранжирование ответов по их релевантности вопросу и выдан один или несколько лучших ответов.

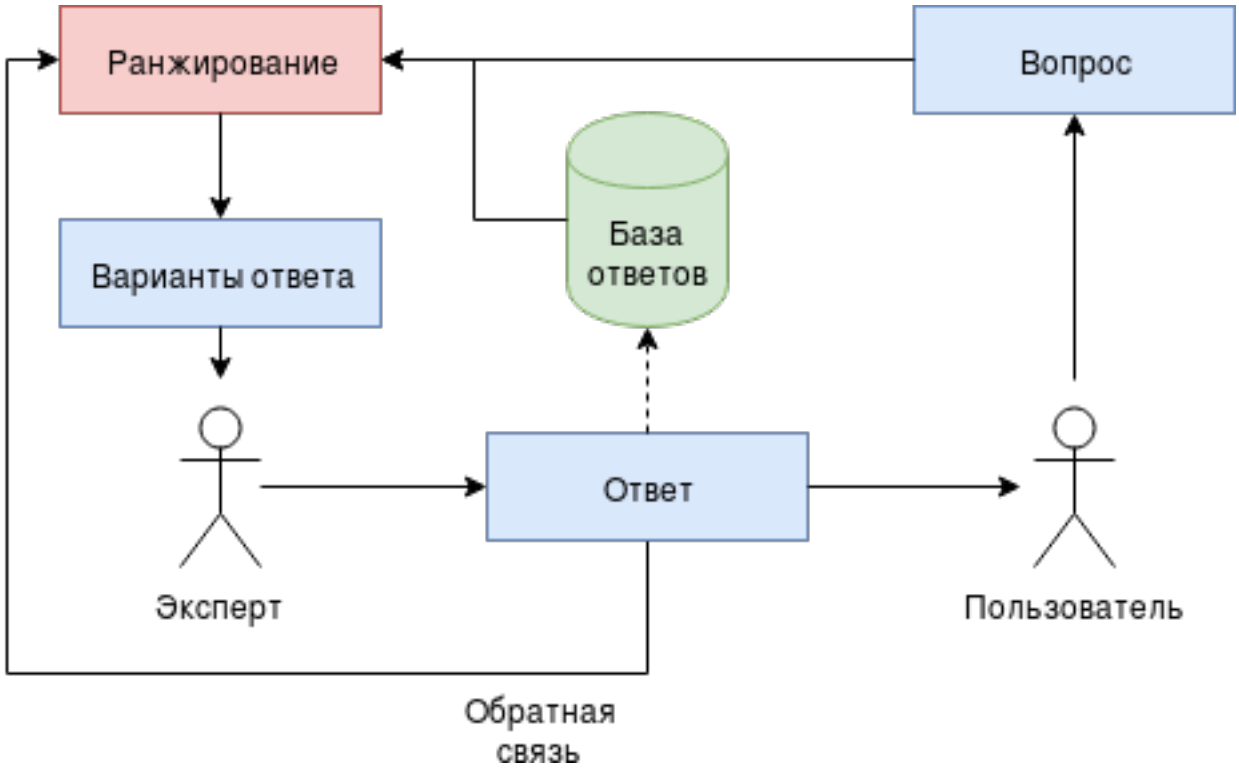

Рис. 1: Схема работы эксперта

Зачастую, во многих задачах присутствует эксперт, а вопросно-ответная система играет роль его помощника, схема работы указана на рис. 1. Это выглядит так: пользователь задает вопрос, вопросно-ответная система выбирает из базы ответов один или несколько вариантов ответа, потом эксперт либо выбирает наиболее подходящий ответ,

либо пишет ответ сам и заносит его в базу ответов. Алгоритм ранжирования должен быть способен подстроиться под изменения в базе ответов и учитывать выбор эксперта, для адаптации к изменениям в области и улучшения качества работы.

1.1 Семинар по оценке вопросно-ответных систем SemEval

Похожие вопросы были рассмотрены на SemEval в 2015[4], 2016[5] и 2017[6] годах. Наиболее схожая задача была в 2017 году.

На SemEval 2017 было 5 подзадач связанных с ранжированием в вопросно-ответных системах. В подзадаче A рассматривалось соответствие вопроса и ответа, в подзадаче B – вопроса и другого вопроса, в подзадаче C – вопроса и ответа, написанного для другого вопроса, в D – вопроса и ответа на арабском, в E – вопроса и другого вопроса на арабском.

Далее для рассмотрения выбрана подзадача A,так как там рассматривается ранжирование ответов в вопросно-ответной системе и представлено множество методов.

На SemEval 2017 в подзадаче A давались вопросы с англоязычных форумов и 10 комментариев размеченных на хорошие и плохие. Участники должны были провести ранжирование этих комментариев согласно тому насколько хорошо комментарии отвечают на поставленный вопрос так, чтоб все хорошие комментарии были выше плохих.

Помимо этого организаторы предоставили простой метод на основе правил: ранг комментария определяется тем, насколько рано он был написан.

1.2 Методы оценки вопросно-ответных систем

Существует множество различных мер качества ранжирования. В данной работе будут рассмотрены меры, которые использовались на SemEval.

1.2.1 Mean average precision

Mean average precision at K – одна из наиболее часто используемых мер качества для ранжирования, помимо этого она является основной мерой на SemEval. Для ее подсчета сначала считается precision at K (pr_K):

$$pr_K = \frac{\sum_{i=1}^{K} r_{true}(e_i)}{K} \tag{1}$$

Где e_i – i-й элемент последовательности, $r_{true}(e)$ – функция равная 1 если e релевантен, 0 иначе.
Минус этой меры заключается в том, что не учитываются позиции правильных элементов (первые или последние).

Далее на основе pr_K считается average precision at K (apr_K):

$$apr_K = \frac{1}{K}\sum_{i=1}^{K} r_{true}(e_i) * pr_i \tag{2}$$

Такая мера учитывает позиции элементов, но качество ранжирования оценивается для отдельно взятого объекта, в то время как их может быть много.

Для того чтоб посчитать Mean average precision at K для N различных объектов вычисляется среднее по apr_K для каждого:

$$map_K = \frac{1}{N}\sum_{i=1}^{N} apr_K^i \tag{3}$$

Где apr_K^i – это apr_K для i-го объекта.
Под MAP подразумевается Mean average precision по всем объектами всем элементам. Эта мера была основной на SemEval 2017.

1.2.2 Mean reciprocal rank

Mean reciprocal rank (MRR) – это еще одна часто используемая мера оценки качества ранжирования. Она представляет собой средним по reciprocal rank (RR_K) для всех объектов, где (RR_K) вычисляется как:

$$RR_K = \frac{1}{\min_{r_{true}(e_k)=1}(k)}, k \in [1...K] \tag{4}$$

Таким образом MRR определяется как:

$$MRR_K = \frac{1}{N}\sum_{i=1}^{N} RR_K^i \tag{5}$$

Особенностью этой меры является то, что она учитывает позицию первого верно предсказанного элемента, игнорируя все остальные.

1.2.3 Average recall

Average recall(AvgRec) – мера, которая позволяет проводить оценку качества классификации. Полнота (recall) для бинарной классификации вычисляется по формуле:

$$recall = \frac{positive_{true}}{positive_{true} + negative_{false}} \tag{6}$$

Где $positive_{true}$ – количество верно предсказанных положительных меток, а $negative_{false}$ – количество не верно предсказанных отрицательных меток.

Average recall позволяет оценивать качество классификации при множестве классов:

$$AvgRec = \frac{1}{N}\sum_{i=1}^{N} recall_i \tag{7}$$

Где N – количество классов, а $recall_i$ - recall по i-му классу.

1.3 Данные

В ходе работы используются данные предоставленные на SemEval 2017. Данные разделены на train1, train2, dev и test наборы. Помимо этого предоставлен размеченные тестовые наборы 2016 и 2015 годов .

Данные представляют из себя наборы обсуждений. В каждом обсуждении есть 1 вопрос и 10 комментариев. Каждый комментарий имеет метку "*good*"(полезный комментарий), "*bad*"(бесполезный комментарий) или "*potentialy usefull*"(потенциально полезный комментарий). Стоит заметить, что при оценке результатов "*potentialy usefull*"считаются так же, как "*bad*". Количество комментариев различных типов не фиксировано, могут быть обсуждения где все комментарии хорошие или и все плохие. Помимо этого в предоставленных данных 2015 года с такая же разметка, но там количество ответов на вопрос не фиксировано. В таблице 1 указано распределение вопросов и различных типов ответов по различным наборам данных.

Набор данных	Вопросы	Ответы	good	potentialy usefull	bad
train1	1411	14110	5287	2461	6362
train2	379	3790	1364	1777	649
train 2015	2480	5971	7418	1504	5971
dev	244	2440	818	1209	413
dev 2015	291	1529	813	172	544
test 2016	327	3270	1329	1941	
test 2017	293	2930	1523	1407	

Таблица 1: распределение данных по корпусам

Помимо самих данных на SemEval 2017 были представлены два базовых решения, а так же инструменты для оценки качества. Предоставленные инструменты оценивают качество работы по трем мерам качества: mean average precision, mean reciprocal rank, average recall.

2 Постановка задачи

Целью данной работы является исследование и разработка существующих методов ранжирования ответов в вопросно-ответной системе. При этом метод ранжирования должен быть способен дообучаться и адаптироваться к изменениям в коллекции ответов. Для достижения данной цели необходимо выполнить следующие задачи:

1. Исследовать существующий методы ранжирования ответов в вопросно-ответной системе, основанные на машинном обучении.

2. Разработать метод ранжирования ответов согласно их релевантности вопросу, который способен дообучаться и адаптироваться к изменениям в коллекции ответов. Разработанный метод должен работать не хуже базового метода предоставленного на SemEval.

3. Провести оценку качества работы разработанного метода на данных с SemEval, с помощью принятых там мер качества (MAP, AvgRec, MRR).

4. Провести анализ влияния различных компонент метода на качество работы.

5. Провести реализацию разработанного метода.

3 Обзор существующих решений

В данной главе приводится описание существующих методов решения подобной задачи. Т.к. наиболее близкая задача была представлена в рамках SemEval, рассмотрены методы оттуда.

Помимо этого описана суть основных идей и концепций, используемых при реализации указанных методов.

3.1 Модели векторного представления слов

Модели векторного представления слов довольно часто используются во многих задачах обработки естественного языка, в том числе в задаче ранжирования ответов в вопросно - ответных системах.

Существует 2 популярных модели для векторного представления слов: CBOW (Continuous Bag-of-Words) и Skip-gram [7]. Word2Vec – это программный инструмент, объединяющий в себе реализацию этих двух моделей.

Плюсом данного подхода к построению моделей является то, что он позволяет работать со словами, отсутствующими в обучающих данных (если они присутствуют в корпусе текстов для обучения Word2Vec) и учитывать близость слов по смыслу, что позволяет получать больше информации для каждого слова.

Кроме того, в 2016 году была предложена модификация этих моделей FastText [8], позволяющая дополнительно учитывать морфологические особенности слов – модель дополнительно строит векторные представления n-грамм, что даёт возможность впоследствии генерировать векторные представления даже для отсутствующих в корпусе для обучения FastText слов.

Помимо этого, существует модель векторного представления GloVe [9]. Она отличается от подхода используемого в Word2Vec подходом к обучению: в Word2Vec представления слов получаются с помощью обработки контекста некоторого слова нейронной сетью, в то время как в GloVe идет обработка матрицы частот появлений слов в контексте целевого слова. Т.е. строится матрица, где для каждого слова (строки) указано как часто оно встречается в контексте других слов (столбец), далее происходит факторизация полученной матрицы для уменьшения размерности и получения векторных представлений слов фиксированного размера.

3.2 Рекуррентные нейронные сети

Во многих решениях задач обработки естественного языка используются рекуррентные нейронные сети так как они способны эффективно обрабатывать последовательную информацию, которой является текст (на уровне слов и на уровне символов).

Основная идея рекуррентных нейронных сетей заключается в том, что информация обрабатывается последовательно и на каждом шаге учитывается результат работы сети на предыдущем шаге.

Было замечено, что при простейшей реализации рекуррентных нейронных сетей могут происходить так называемые затухания и взрывы градиентов: в пером случае при обучении нейронной сети градиенты приобретают малые по модулю значение и слабо влияют на первые слои сети, во втором – градиенты принимают большие по модулю значение, что может негативно сказаться на обучении или даже вызвать переполнение параметров.

Наиболее популярной модификаций этого подхода являются Long Short Term Memory(LSTM) сети [10], предполагается что они позволяют избежать данных проблем.

LSTM нейронная сеть состоит из ячеек определяющихся следующим образом:

$$f_t = \sigma(W_f \cdot [h_{t-1}, x_t] + b_f) \tag{8}$$

$$i_t = \sigma(W_i \cdot [h_{t-1}, x_t] + b_i) \tag{9}$$

$$S_t = \tanh(W_S \cdot [h_{t-1}, x_t] + b_S) \tag{10}$$

$$C_t = f_t \odot C_{t-1} + i_t \odot S_t \tag{11}$$

$$o_t = \sigma(W_o \cdot [h_{t-1}, x_t] + b_o) \tag{12}$$

$$h_t = o_t \odot \tanh(C_t) \tag{13}$$

Где W и b – обучаемые параметры, общие для всех ячеек, а $\odot$ – поэлементное произведение. C_t – ячейка памяти, где хранятся знания с предыдущих шагов. f_t и i_t предназначе-

ны для частичного забывания старой информации и фильтрации новой соответственно, в то время как o_t контролирует долю выдаваемой информации.

LSTM при работе делает проход в одном направлении, но во многих задачах обработки текстов важен не только левый, но и правый контекст, поэтому часто используют двунаправленные LSTM (Bi-LSTM), как например в [11]. Идея данного подхода заключается в том, что последовательность обрабатывается двумя разными LSTM, которые читают ее в прямом и обратном порядке. Далее их выходы конкатенируются и могут быть использованы в дальнейшем.

3.3 Модель KeLP

Одно из лучших решений на SemEval 2017 в подзадаче A предоставила команда KeLP [12]. Они разработали метод на основе SVM с различными ядрами и множеством признаков.

3.3.1 Способ ранжирования

Из вопросов и ответов составляются пары, а из полученных пар извлекаются признаки, которые будут описаны далее. Релевантным ответам ставится в соответствие положительная метка, не релевантным – отрицательная.

С помощью SVM с использованием полученных признаков проводится бинарная классификация соответствие вопроса и ответа. Ранжирование проводится соответственно результату работы SVM.

3.3.2 Ядра

В KeLP применяются ядра для деревьев разбора и линейные ядра для признаков попарного сходства и специфичных признаков, которые описаны далее.

По вопросу и ответу строятся деревья их синтаксического разбора по грамматике составляющих, далее обрезаются наименее информативные вершины и к полученным деревьям применяются Partial Tree Kernel (PTK) [13]. Это позволяет извлекать из предложений их синтаксическое сходство.

3.3.3 Парные сходства

Предполагается, что вопрос и релевантный ему ответ содержат много общих элементов. Для оценки их схожести используются различные метрики:
Cosine similarity: для двух векторов a, b вычисляется косинус угла между ними

$$CosSim = \frac{a \cdot b}{||a|| \cdot ||b||} \tag{14}$$

Jaccard coefficient: для двух множеств A, B jaccard coefficient вычисляется по следующей формуле

$$J(A, B) = \frac{|A \cap B|}{|A \cup B|} \tag{15}$$

Эти и многие другие метрики используются для улавливания различных типов схожести вопроса и ответа:

- **Лексическая схожесть**: Cosine similarity, Jaccard coefficient и containment measure по n-граммам лемм слов, длина наибольшей общей подстроки, длина наибольшей общей подпоследовательности и Greedy String Tiling.
- **Синтаксическая схожесть**: Cosine similarity по n-граммам меток частей речи (POS)
- **Семантическая схожесть**: Cosine similarity по усредненным векторным представлениям word2vec для существительных, глаголов, прилагательных, наречий и всех слов из текста.

3.3.4 Специфичные признаки

Все ранее описанные признаки характеризуют схожесть текстов вопроса и ответа. Помимо этих признаков были добавлены специфичные признаки, разработанные для задачи ранжирования ответов по их релевантности вопросу применительно к случаю форумов. Для этого используются следующие признаки:

- **Признаки ранжирования**: использовались результаты работы алгоритмов инфопоиска на множестве комментариев.

- **Эвристики**: используются эвристические признаки описанные в [14] чтоб получить характеристику ответа, такие как его длинна, тип комментария, содержание ссылок, ключевых слов и пр.

- **Признаки форума**: комментарии, которые не отвечают на вопрос, а являются частью дискуссии отдельно помечаются.

3.4 Модель Beihang-MSRA

Так же высокие результаты получил на SemEval 2017 в подзадаче A метод Beihang-MSRA [15], который представляет собой комбинацию из нейронной сети и XGBoost регрессии.

3.4.1 Способ ранжирования

Из вопросов и ответов составляются размеченные пары. Нейронная сеть используется для генерации вектора, описывающего пару текстов на основе последовательностей их слов. Она обучатся бинарной классификации релевантности ответа вопросу. Полученный вектор комбинируется с дополнительными признаками и используется для решения задачи регрессии с помощью XGBoost. На основе результатов производится ранжирование.

3.4.2 Подготовка

К текстам вопроса и ответа применяется разбиение текста по словам (токенизация), отсечение окончаний (стемминг), удаление не информативных слов и приведение всех слов к нижнему регистру. Помимо этого происходит расстановка меток частей речи (POS) и построение дерева синтаксического разбора.

3.4.3 Традиционные признаки

В решении используются признаки основывающиеся на словах и синтаксическом анализе.

- **Tf-idf cosine**: тексты векторизуются с Tf-idf взвешиванием: для каждого слова из текста считается term frequency (tf) – отношение числа вхождений некоторого

слова к общему числу слов документа и inverse document frequency (idf) – инверсия частоты, с которой некоторое слово встречается в документах коллекции для них считается произведение $tf - idf = tf * idf$. Считается cosine similarity между полученными представлениями

- **Longest common subsequence**: считается длина наибольшей общей подпоследовательности токенов текстов и нормализуется максимумом длин последовательностей.

- **Word overlap**: вычисляется количество общих n-грамм и существительных нормализованное максимальным количеством n-грамм и существительных среди текстов.

- **Tree kernels**: используются для измерения синтаксической близости пары текстов. По дереву синтаксического разбора считаются subtree kernel [16], subset tree kernel [17] и partial tree kernel [13].

- **Translation probability**: с помощью инструмента GIZA++ [18] вычисляется вероятность перевода вопроса в ответ, где вопрос считается исходным языком, а ответ – целевым.

- **Word embedding cosine**: для вопроса и ответа соответственной вычисляется среднее по векторным представлениям их слов. Для полученных векторов вычисляется cosine similarity в качестве признака.

3.4.4 Специфичные признаки

Помимо, указанных выше, признаков были предложены дополнительные, предназначенные специально для поставленной задачи. Эти признаки основываются на эвристических наблюдениях о качестве ответа.

- Является ли автор комментария автором вопроса

- Длина комментария

- Наличие ссылок или электронных адресов

- Наличие позитивных или негативных смайликов.

3.4.5 Нейронная сеть

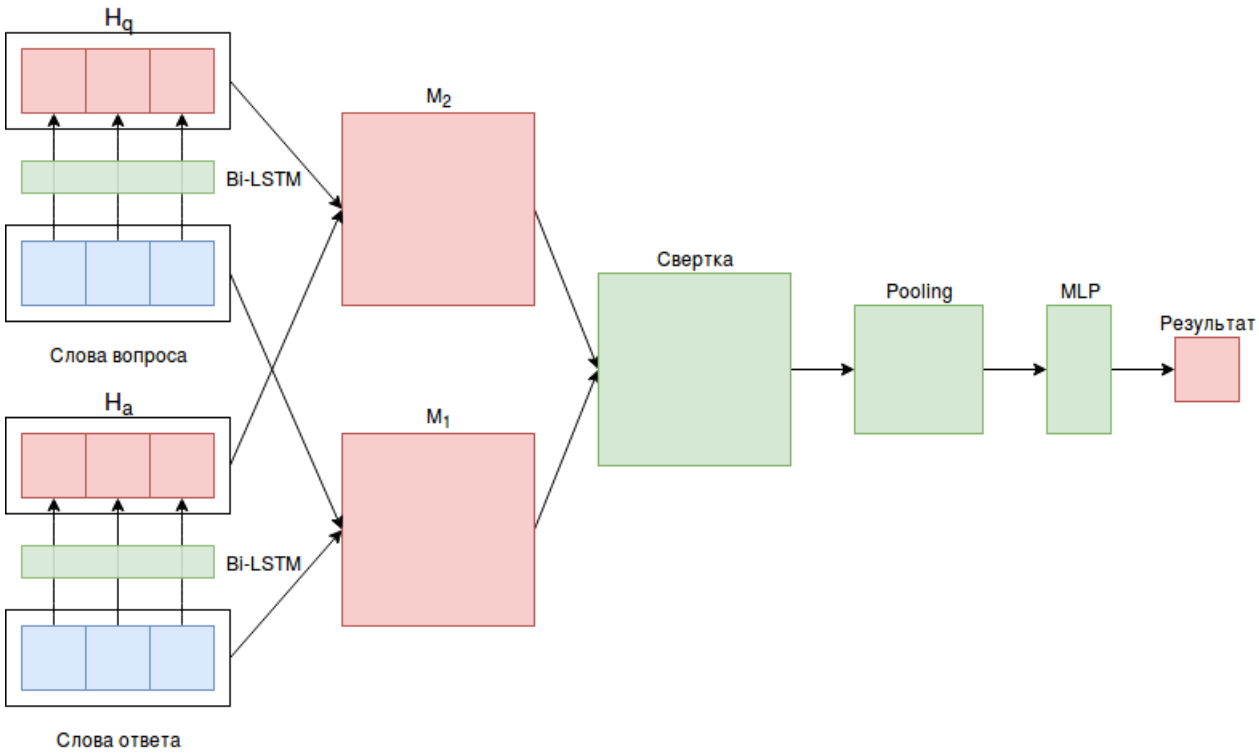

Рис. 2: Архитектура нейронной сети у Beihang-MSRA

На вход нейронной сети подаются две последовательности векторных представления S_q длины I и S_a длины J для вопроса и ответа соответственно. Каждая последовательность обрабатывается с помощью Bi-LSTM и на выходе получаются последовательности H_q и H_a.

С помощью имеющихся последовательностей вычисляются 2 матрицы: $M_1 = [m_{1,i,j}]_{I,J}$ и $M_2 = [m_{2,i,j}]_{I,J}$. Где M_1 задается следующим образом: $m_{1,i,j} = s_{q,i}^T \cdot s_{a,j}$, она представляет схожесть слов вопроса и ответа. M_2 задается как: $m_{2,i,j} = h_{q,i}^T A h_{a,j}$, где A – обучаемая матрица весов.

Далее к полученным матрицам применяется двухмерная свертка с max pooling и relu активацией. Матрицы M_1 и M_2 считаются двумя каналами. Полученные вектора для вопроса и ответа конкатенируются и далее передаются в перцептрон где в итоге решается задача классификации на релевантность ответа вопросу. В качестве функции

потерь используется бинарная кросс энтропия.

$$cost(y_{true}, y_{pred}) = -y_{true}\log(y_{pred}) - (1 - y_{true})\log(1 - y_{pred}) \tag{16}$$

Стоит заметить, что для обучения нейронной сети были использованы не только данные с SemEval, но и данные, которые были получены из Yahoo Answer, чтоб избежать переобучения нейронной сети.

3.5 Модель FuRongWang

Еще один метод на SemEval 2017 показала команда FuRongWang[19]. Их метод полностью основан на нейронной сети, проводящей классификацию на релевантность ответа вопросу.

3.5.1 Способ ранжирования

Из вопросов и ответов составляются пары. Извлекаются word2vec представления слов и дополнительные признаки. По полученным последовательностям строятся представления для вопроса и ответа либо с помощью рекуррентной сети, либо с помощью сверточной сети. В дальнейшем это используется для бинарной классификации ответа на релевантность вопросу. Ранжирование проводится по выданным нейронной сетью вероятностям релевантности.

3.5.2 Архитектура нейронной сети

На рисунке 3 проиллюстрирована архитектура нейронной сети в данном методе. В данном подходе для обработки последовательностей признаков слов может быть использовано 2 архитектуры:

1. **Рекуррентная**: последовательности передаются в bi-LSTM и конкатенация выходов прямой и обратной LSTM передается дальше.

2. **Сверточная**: каждая последовательность либо обрезается, либо расширяется до 200 элементов. Далее применяется двухмерная свертка с max_pooling. Полученный вектор передается дальше.

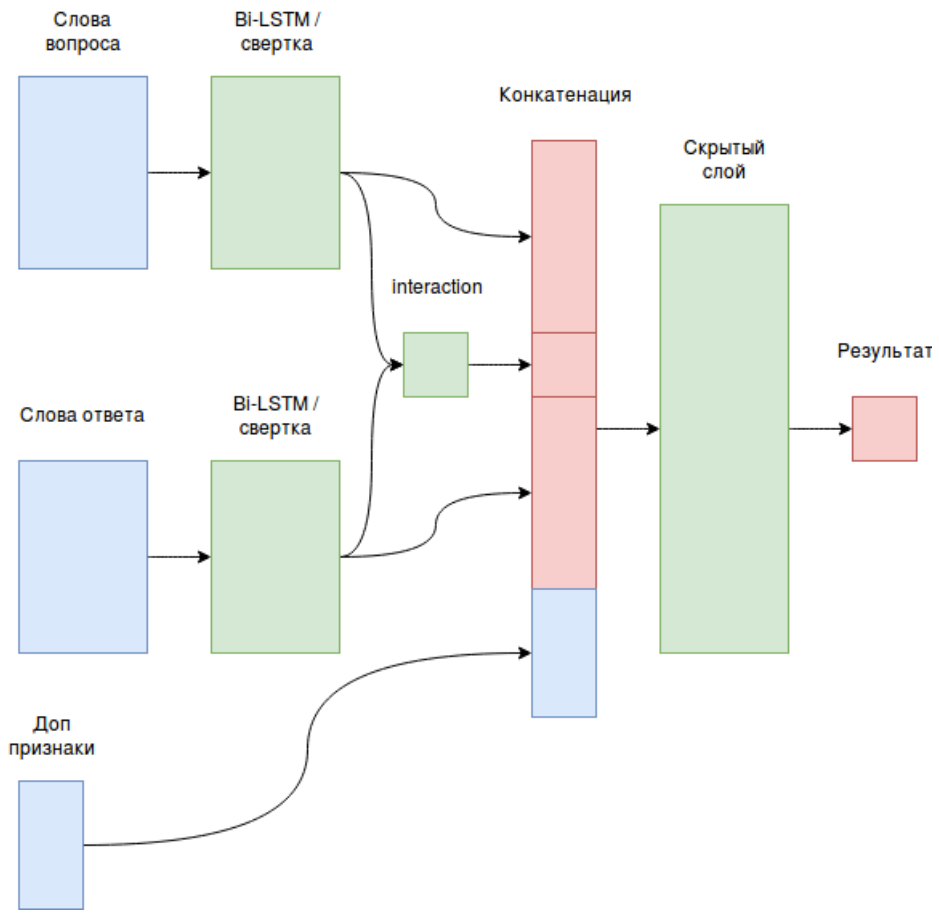

Рис. 3: Архитектура нейронной сети у FuRongWang

Вектора, полученные в результате обработки признаков слов предаются в Interaction layer, описанный далее. После этого они конкатенируются с его выводом, а так же с дополнительными признаками. Далее стоит полносвязный слой с relu активацией и sigmoid слой. В качестве функции потерь используется бинарная кросс энтропия.

3.5.3 Interaction Layer

В данном методе используется так называемый interaction layer. Он определяется следующим образом: пусть $q \in \mathbb{R}^d$ – вектор вопрос , $a \in \mathbb{R}^d$ – вектор ответа, W – матрица весов размера $[d, d]$, а $f()$ – функция активации. Тогда interaction layer вычисляется следующим образом:

$$int = f(q^T W a) \tag{17}$$

Interaction layer полезен тем, что может вычислить схожесть между представлениями вопроса и ответа с учетом связей между всеми элементами, в отличии от cos similarity, который учитывает только элементы на соответствующих позициях.

3.5.4 Признаки

В качестве признаков слов используются их word2vec представления. Помимо этого используется бинарный признак пересечения по словам: он принимает значение 1, если слово входит в другую последовательность, иначе 0.

Признаки пересечения слов конкатенируются и используются с полученными представлениями вопроса и ответа для классификации. Этот признак может быть использован с предложениями любой длинны, так как на них задается ограничение в 200 элементов и предложение либо обрезается, либо добавляются нулевые элементы.

3.6 Выводы

Была рассмотрена задача с SemEval, которая является наиболее близкой к задаче поставленной в данной работе.

Рассмотрены методы, представленные на SemEval 2017. Они используют множество подходов и признаков. Их можно использовать при разработке собственного метода, однако признаки, завязанные на специфике форумов, не соответствуют ограничениям задачи.

Стоит заметить, что SVM не предоставляет возможность дообучения модели, а метод на основе комбинации нейронной сети и XGBoost плохо дообучается, так как он состоит из двух компонентов, которые связанны не напрямую: при дообучении нейронной сети поменяется ее вывод и надо будет обучать XGBoost с нуля, а если дообучать только XGBoost, то нейронная сеть устареет. Так что при построении метода нельзя их брать за основу.

4 Исследование и построение решения задачи

В данной главе описывается разработанный метод ранжирования ответов в вопросно-ответной системе. При конструировании метода были использованы как идеи, позаимствованные из уже существующих методов, так и некоторые экспериментальные идеи.

Изначально сконструировано базовое решение, далее к нему добавлялись различные признаки и небольшие изменения в архитектуре. Ниже приведен анализ результатов работы в зависимости от добавления признаков или изменений в архитектуре.

4.1 Оценка качества

Финальная оценка качества производилась на тестовых данных с SemEval 2017 с помощью предоставленных там инструментов оценки. По аналогии с командой FuRongWang в данной работе для обучения используются train1 + train2 + train2015, для подбора параметров dev + dev2015 + test2016, общие результаты вычисляются на test2017.

4.2 Архитектура нейронной сети

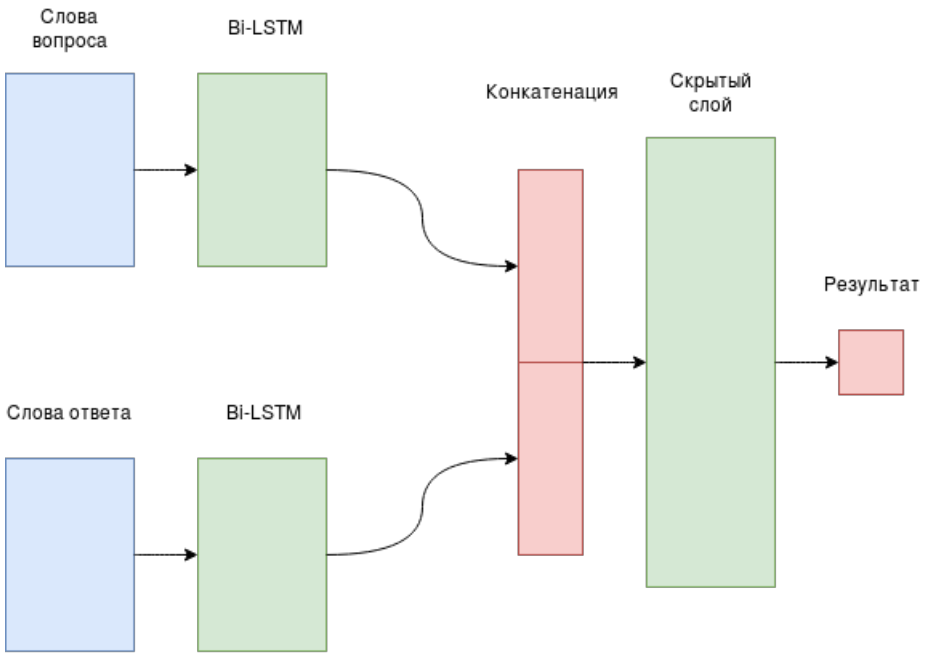

Рис. 4: Базовая архитектура нейронной сети

В качестве решения используется нейронная сеть, проводящая классификацию на релевантность ответа вопросу. Она принимает 2 типа признаков: признаки слов, комбинация которых передается в виде последовательности и общие признаки, которые добавляются после обработки признаков слов. Архитектура нейронной сети изображена на рис. 4.

Признаки слов для вопроса S_q и ответа S_a обрабатываются с помощью Bi-LSTM с dropout на входе. В итоге получаются векторы h_q и h_a. Далее они комбинируются с общими признаками c, и комбинация $[h_q, h_a, c]$ проходит еще один скрытый слой и далее переходит на выходной с $sigmoid$ в качестве активации.

4.3 Обучение нейронной сети

В качестве функции потерь используется бинарная кросс энтропия с регуляризацией. В качестве алгоритма обучения используется adam [20], т.к. он быстро сходится к минимуму, за счет адаптивного подбора коэффициента, на который умножаются градиенты.

Так же проводится усечение градиентов с большой нормой, описанное в [21], для того чтоб избежать резких изменений параметров нейронной сети и взрывов градиентов. Данные подаются частями, так что каждой части соответствует 1 вопрос и все комментарии, относящиеся к нему, т.е. одно обсуждение из коллекции данных.

4.4 Признаки

В разработанном методе признаки можно условно разделить на 2 типа: признаки слов и общие признаки. Признаки слов относятся только к одному конкретному слову и тексты вопроса и ответа описываются последовательностью признаков слов. В свою очередь общие признаки характеризуют свойства текста целиком или свойства пары текстов вопроса и ответа.

4.4.1 Признаки слов

В первую очередь используется модель векторного представления слов. Данное метод может работать как и с моделью word2vec, так и с моделью fasttext или GloVe. Стоит отметить, что модели векторного представления слов так же могут отличаться размерностью и данными, на которых они обучены.

Помимо векторных представлений, могут быть использованы POS слов. Для этого предварительно проводится расстановка POS. POS кодируются в виде one-hot векторов (вектор с 1 на определенной позиции, и 0 на остальных).

4.4.2 Общие признаки

Были выбраны признаки, которые указывают на схожесть между двумя текстами, из описанных ранее методов.

- **Word embedding cosine**: это признак вычисляется так же как и в [15], предполагается что он указывает на семантическую близость текстов.
- **Longest commom subsequence**: этот признак тоже был описан в [15], предполагается что он указывает на синтаксическую близость текстов за счет количества повторяющихся терминов.
- **Overlap**: этот признак вычисляется аналогично признаку пересечения последовательностей в [19], предполагается что он указывает на синтаксическую близость текстов. При включении этого признака, входные последовательности ограничиваются 200 элементами.

4.5 Модификации архитектуры

Помимо признаков, большой вклад в результат работы метода вносит способ их обработки. Далее описаны различные модификации базовой архитектуры сети, которые теоретически могут улучшить результат работы решения.

- **Interaction**: для сравнения полученных представлений для вопроса и ответа, может быть использован interaction layer, по аналогии с [19]. Эта модификация изображена на рис. 5.

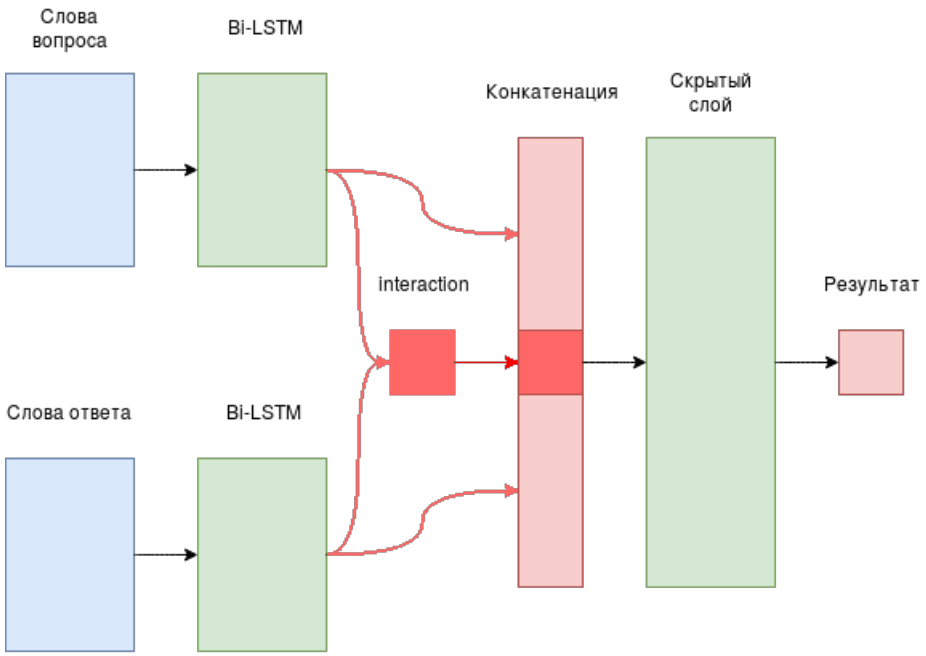

Рис. 5: Архитектура нейронной сети с interaction

- **Attention**: после работы каждого LSTM можно использовать attention механизм [22]. Он позволяет обращать внимание только на наиболее важные элементы последовательности, игнорируя не нужные. Архитектура с этой модификацией изображена на рис .6.

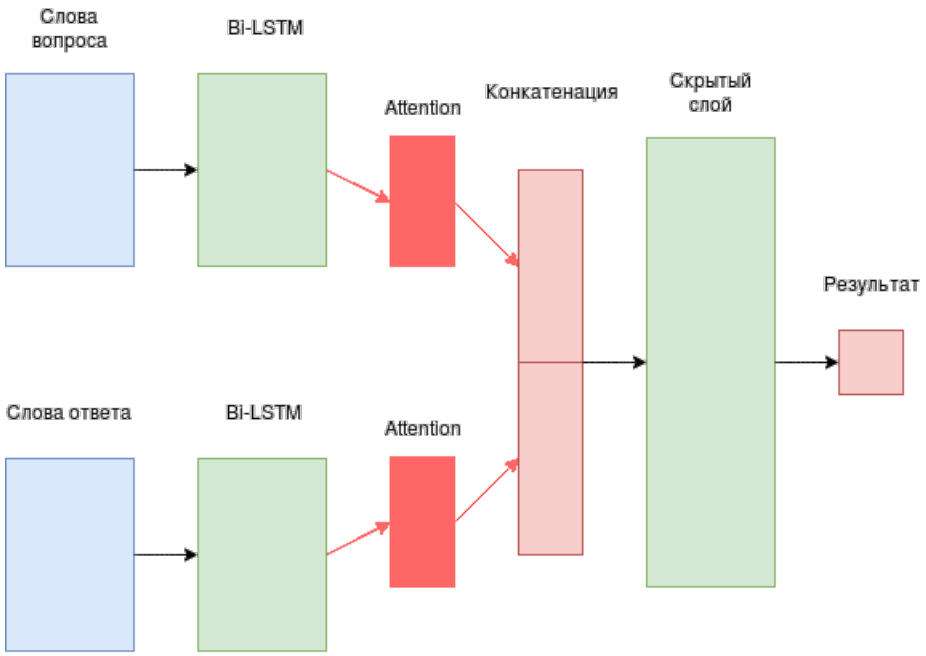

Рис. 6: Архитектура нейронной сети с attention

- **Использование вектора вопроса в качестве признака для построения ответа**: можно попробовать учитывать контекст при построении вектора ответа. Для этого можно использовать вектор вопроса и передавать его в качестве признака на каждом шагу последовательности ответа. Архитектура с данной модификацией изображена на рис. 7.

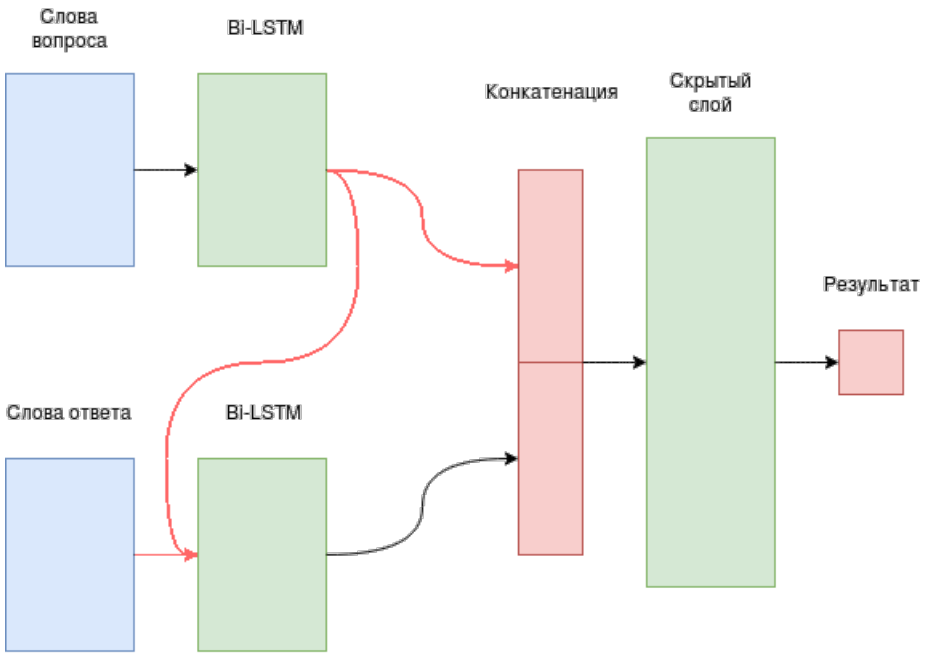

Рис. 7: Архитектура нейронной сети с использованием вектора вопроса

- **Char CNN**: Для получения дополнительной информации о словах может быть использован подход с использованием Char CNN: для всех символов из обучающей выборки создаются случайные векторные представления, и выполняется свертки по символам слова. Векторные представления символов улучшаются по мере обучения. Базовая архитектура с данной модификацией изображена на рис. 8.

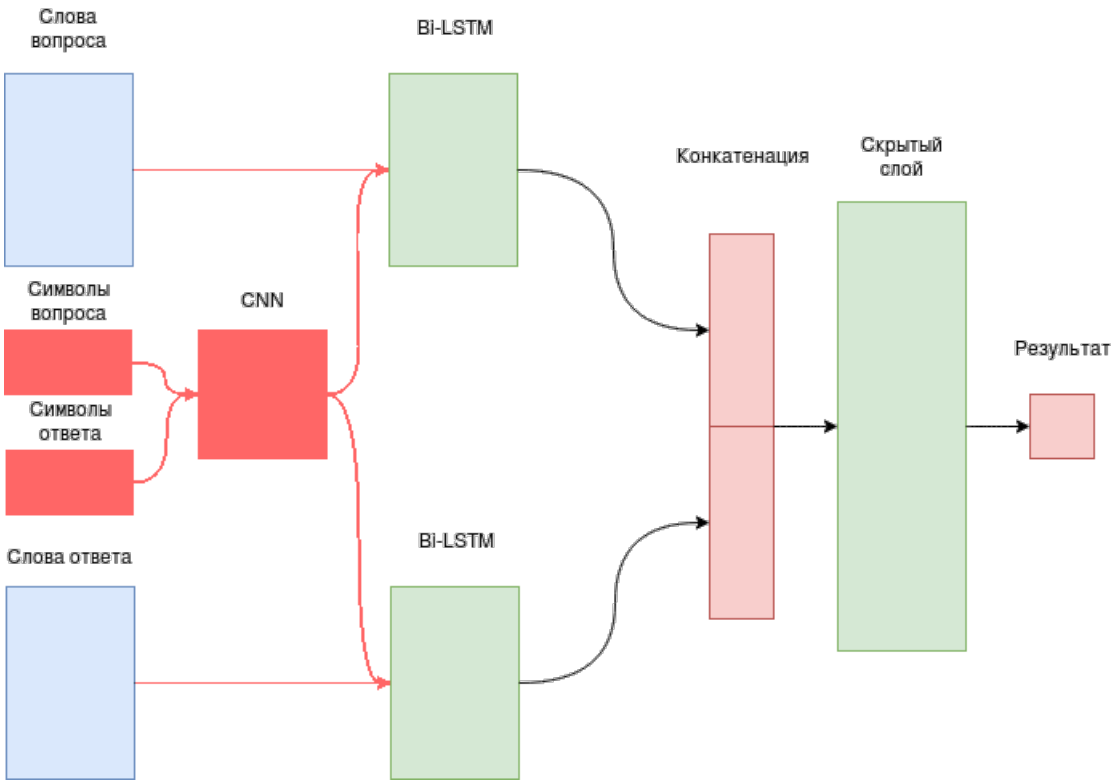

Рис. 8: Архитектура нейронной сети с использованием Char CNN

4.6 Эксперименты

В данной части описаны результаты обучения моделей для решения задачи и приведено сравнение различных конфигураций решения задачи. Для каждого эксперимента проведено 3 запуска, так как результат обучения нейронной сети зависит от ее инициализации. Оптимальные параметры и эпоха выбираются на dev 2015 + dev 2016 + test 2016, результат считается на test 2017. Далее приводятся усредненные результаты.

4.6.1 Сравнение векторных представлений

В первую очередь необходимо выявить наиболее эффективную модель векторных представлений слов для данной задачи. Помимо самой модели, существенное влияние оказывают данные, на которой модель обучена.

В ходе работы были собраны текстовые данные из английской википедии и из обучающей выборки предоставленной на SemEval 2017. На этих данных обучены три модели векторного представления слов: Word2Vec, FastText и GloVe. Размерность векторов для

каждой модели равна 100, при обучении FastText и GloVe на википедии брались слова которые хотя бы 20 раз встречались в тексте, на данных с SemEval – хотя бы 3.

С каждой моделью проведено тестирование базовой конфигурации метода. Помимо них проведено тестирование модели GloVe предоставленной авторами данной модели, которая была обучена на данных с википедии и корпусе Gigaword 5.

Результаты тестирование представлены в таблице 2

Модель	Dev			Test		
	MAP	AvgRec	MRR	MAP	AvgRec	MRR
Stanford GloVe	69.45	80.20	75.52	72.01	80.17	81.64
Wiki GloVe	69.46	80.20	75.54	71.82	80.76	81.65
SemEval GloVe	69.51	80.15	75.67	71.96	80.66	81.51
Wiki FastText	77.04	**85.98**	**81.89**	84.15	90.93	89.21
SemEval FastText	**77.14**	85.89	81.43	**84.97**	**91.29**	**89.40**
Wiki Word2Vec	69.57	80.24	75.51	72.49	81.14	82.02
SemEval Word2Vec	69.66	80.36	75.55	72.67	81.33	81.94

Таблица 2: Результаты работы базового решения

Как можно заметить, модели FastText серьезно превосходят остальные. Это может быть связано с тем, что FastText позволяет строить векторные представления для слов, которые никогда ранее не встречались.

Помимо этого, заметно то, что результаты работы с моделями, обученными на текстах из тренировочных данных, часто превосходят результаты работы с остальными моделями. Это может быть связанно с тем, что на форумах используется лексикон, который специфичен для них и отличается от лексикона, используемого в википедии.

4.6.2 Подбор оптимальной архитектуры

Далее проведено исследование модификаций архитектуры. Результаты описаны в таблице 3.

Модификация	Dev			Test		
	MAP	AvgRec	MRR	MAP	AvgRec	MRR
Baseline	77.14	85.89	81.43	84.97	91.29	89.40
+ Attention	**77.33**	**86.20**	**82.60**	85.53	91.95	90.27
+ Interaction	77.11	86.12	82.23	85.43	91.97	89.61
+ Q2A.	77.31	86.02	82.21	86.02	92.14	90.89
+ Char	77.11	85.97	81.91	85.82	92.24	90.55
+ Char + Q2A	77.26	86.13	82.23	**86.19**	**92.34**	**91.25**

Таблица 3: Модификации нейронной сети

Как видно, на dev лучше всего показала себя модификация с Attention, и при добавлении каких либо еще модификаций, результат на dev падает. Таким образом лучшей модификацией признается добавление Attention механизма.

4.6.3 Дополнительные признаки

Помимо этого, были опробованы различные дополнительные признаки. Результаты указаны в таблице 4.

Признак	Dev			Test		
	MAP	AvgRec	MRR	MAP	AvgRec	MRR
Baseline	77.33	**86.20**	**82.60**	**85.53**	91.95	90.27
+ Word embedding cosine	**77.43**	86.11	82.26	85.46	91.93	89.86
+ POS	77.26	85.94	82.11	85.28	91.83	89.69
+ Longest common subsequence	77.25	85.96	82.19	85.41	**92.06**	**90.47**
+ Overlap	77.18	85.88	81.93	84.59	91.20	89.29

Таблица 4: Дополнительные признаки

Как видно из таблицы, на dev лучший результат по MAP показало добавление word embedding cosine. По остальным признакам лучше всего себя показало базовое решение с Attention и Fasttext обученном на тренировочных данных. Для финальной версии

выбирается архитектура с добавлением word embedding cosine, так как MAP являлась основной метрикой на SemEval

4.6.4 Сравнение с существующими решениями

По результатам экспериментов были выбраны модификации и признаки, дающие наибольший прирост по MAP на dev. Архитектура итоговой нейронной сети представлена на рис. 9.

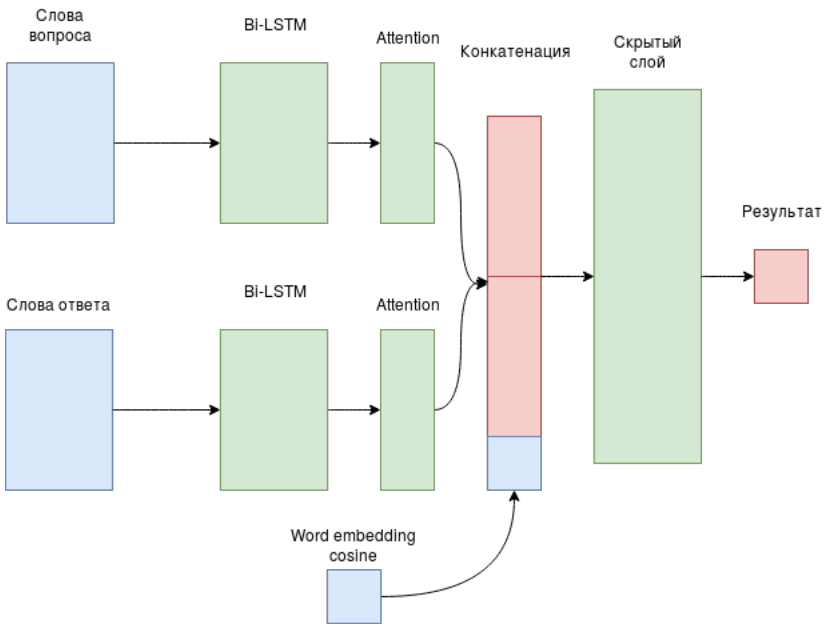

Рис. 9: Итоговая архитектура нейронной сети

Было проведено сравнение с описанными на SemEval решениями и предоставленными там базовыми решениями. Результаты указаны в таблице 5.

Метод	MAP	AvgRec	MRR
KeLP [12]	88.43	93.79	92.82
Beihang-MSRA [15]	88.24	93.87	92.34
Данная работа	85.46	91.93	89.86
FuRongWang [19]	84.26	90.79	89.40
chronological baseline [6]	72.61	79.32	82.37
random baseline [6]	62.30	70.56	68.74

Таблица 5: Сравнение с существующими решениями

Где chronological baseline – базовое решение ранжирующее ответы по тому, насколько рано их написали, random baseline – базовое решение, проводящее случайное ранжирование ответов.

4.7 Выводы

По результатам экспериментов были выбраны признаки и модификации для итоговой архитектуры нейронной сети. Полученное решение работает хуже чем у Beihang-MSRA [15] и KeLP [12], однако они не удовлетворяют условиям задачи: метод KeLP не позволяют проводить дообучение, так как он основан на SVM, а метод Beihang-MSRA состоит из двух независимых частей, что не позволяет дообучать их одновременно. В то время, как оно превзошло решение, удовлетворяющее постановке задачи.

Базовая архитектура нейронной сети и все соответствующие модификации были реализованы. Помимо этого реализованы механизмы извлечения признаков и обучены модели векторного представления слов с помощью разных алгоритмов, с использованием текстов из английской wiki и текстов из данных предоставленных на SemEval 2017.

5 Описание практической части

5.1 Обоснование выбранного инструментария

Основная часть работы выполнена на языке Python. Этот язык довольно прост, благодаря чему с его помощью можно легко и быстро создать прототип решения. Помимо этого, для Python существует множество библиотек с инструментами для машинного обучения и обработки текстов.

Для создания нейронной сети использовалась библиотека Tensorflow, так как она позволяет просто конструировать и обучать нейронные сети. Помимо этого Tensorflow позволяет проводит вычисления как на CPU так и на GPU, и она работает быстро.

Для токенизации и извлечения POS-тегов использовалась Texterra. Эта библиотека предоставляет широкие возможности для обработки текстов на естественном языке, работает эффективно и имеет хорошую и понятную документацию.

Для работы с векторными представлениями использовались библиотеки Gensim и fasttext. С их помощью проводилось чтение и обучение векторных представлений. Помимо этого использовались специализированные инструменты для обучения GloVe.

5.2 Общая схема работы

Схему работы данного решения можно разбить на 2 части: обучение и тестирование. При обучении:

1. Загружается модель векторного представления слов, а так же данные для обучения в формате xml и файл конфигурации в формате json.

2. Из данных извлекаются вопросы и комментарии с метками.

3. Тексты для них токенизируются.

4. Из отдельных слов извлекаются признаки уровня слова, а так же извлекаются общие признаки для пары текстов, которые были указаны в конфигурационном файле (например: векторные представления).

5. Строится структура нейронной сети согласно конфигурации.

6. Нейронная сеть обучается на полученных данных в течении нескольких эпох. На каждой эпохе сохраняется модель нейронной сети.

Схема работы при обучении показана на рис. 10.

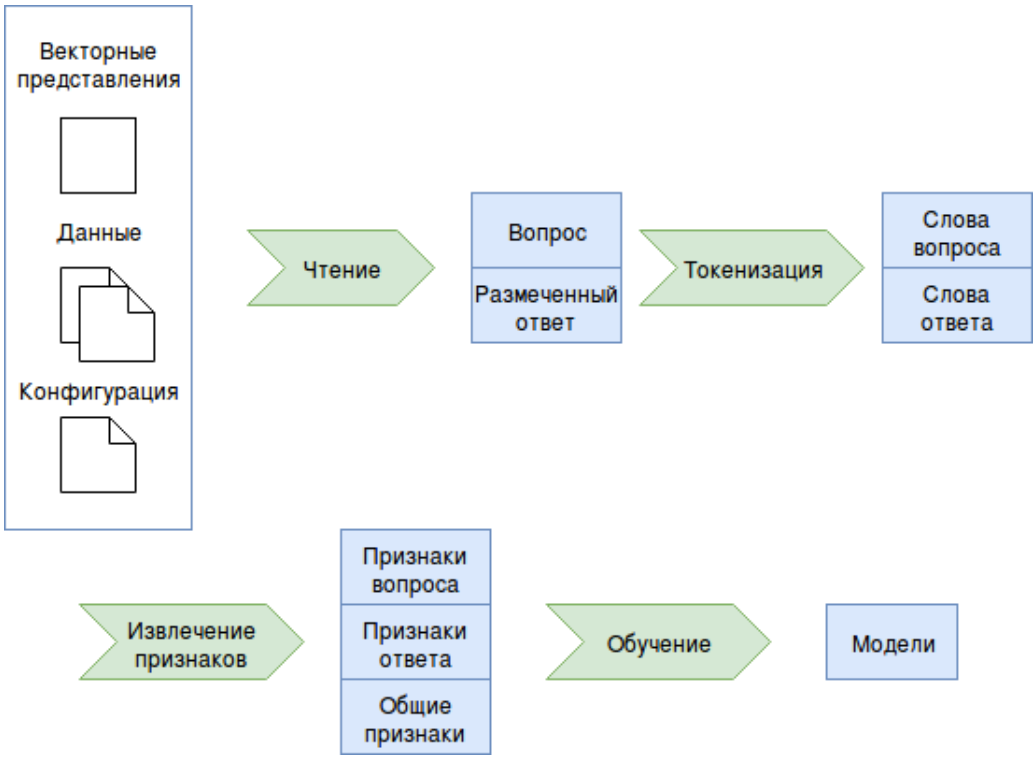

Рис. 10: Схема работы при обучении

При тестировании:

1. Загружается модель нейронной сети, модель векторного представления слов, данные и конфигурация.

2. Из данных извлекаются признаки аналогично пунктам 2-4 обучения.

3. В нейронную сеть для предсказания подаются признаки пар вопросов и ответов, сгруппированных по вопросам. Выданное число помечается как ранг ответа.

4. Полученные результаты сохраняются в специальном формате, поддерживаемом скриптом тестирования SemEval.

5. Данным скриптом проводится оценка качества.

Схема работы при тестировании показана на рис. 11.

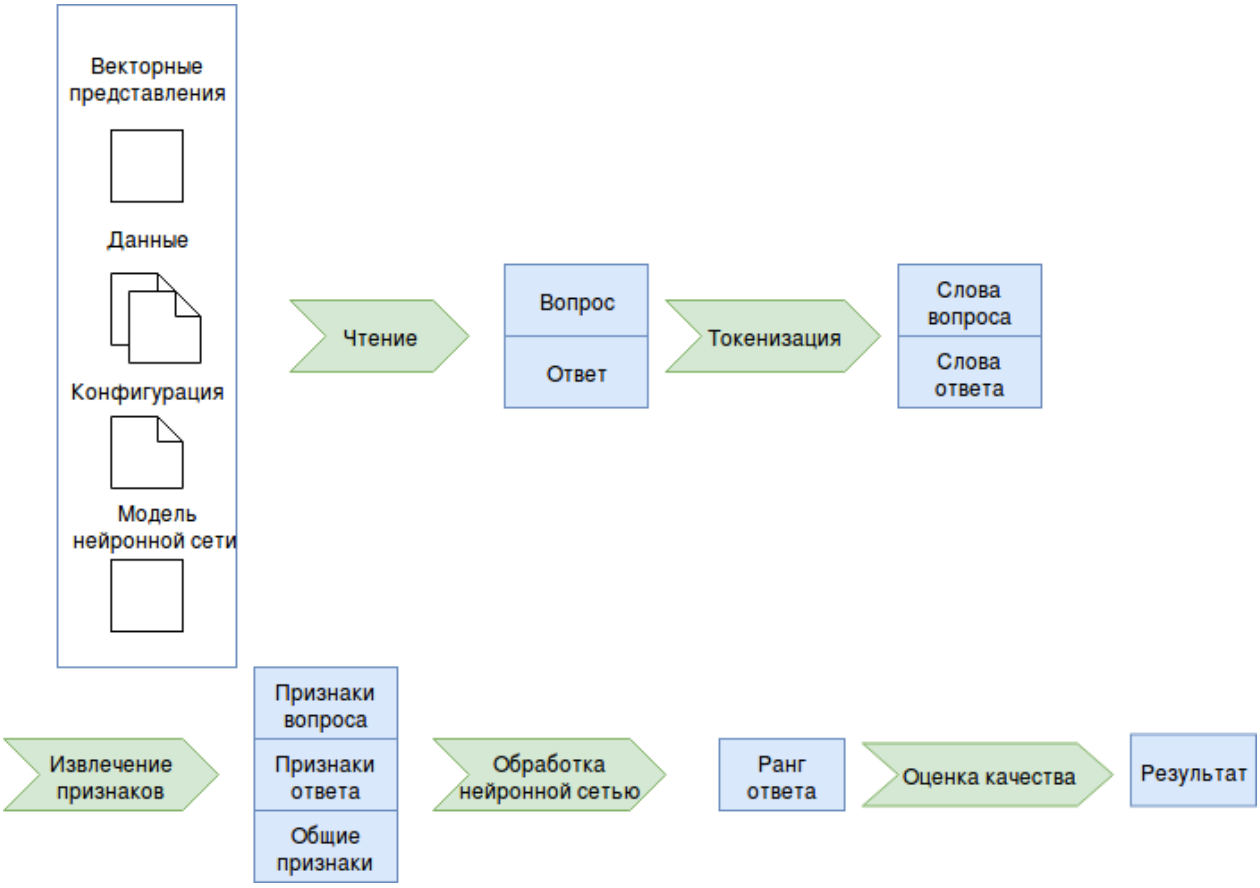

Рис. 11: Схема работы при тестировании

5.3 Общая архитектура системы

Система состоит из следующих основных модулей:

- **FeatureExtractor**: обрабатывает список обсуждений и извлекает из них признаки, загружает модель векторного представления при инициализации.
- **Classifier**: инициализирует нейронную сеть, дает возможность ее обучать, предсказывать, и так же сохранять или загружать ее.
- **Data reader**: может читать данные с SemEval в формате XML и записывать результат обработки.

- **Trainer**: инициализирует FeatureExtractor и Classifier, проводит чтение данных с помощью Data reader и обучение нейронной сети и сохраняет ее.
- **Ranker**: инициализирует FeatureExtractor и Classifier, проводит чтение данных с помощью Data reader. С помощью Classifier проводит ранжирование с записью результатов.

На рис. 12 показана схема взаимодействия описанных модулей.

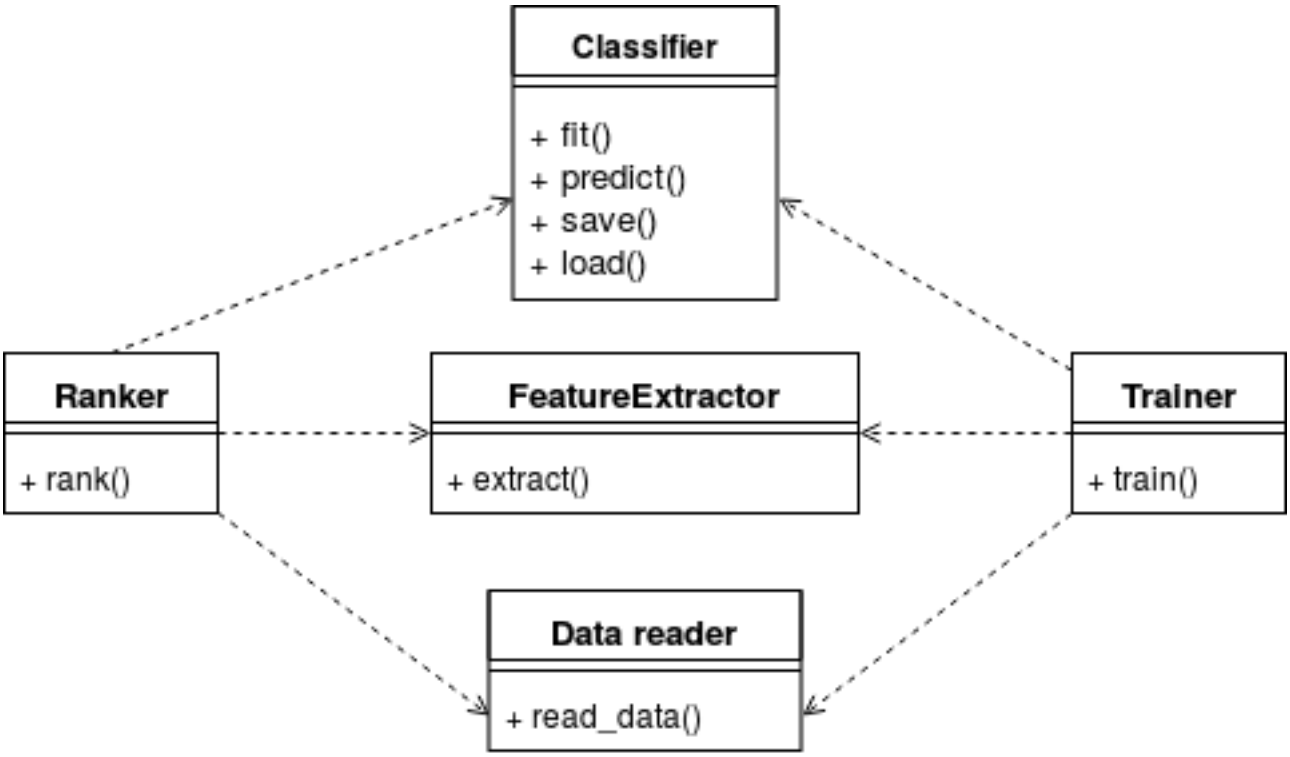

Рис. 12: Схема взаимодействия модулей

Заключение

По итогам данный работы было проведено исследование существующих методов ранжирования ответов согласно релевантности вопросу и реализован собственный метод. Были выполнены следующие подзадачи:

1. Исследованы существующие методы ранжирования ответов в вопросно-ответной системе.

2. Разработан метод ранжирования ответов по релевантности вопросу, который способен до обучаться и адаптироваться к изменениям в коллекции ответов. Разработанный метод работает не хуже базового метода предоставленного на SemEval.

3. Выполнена реализация базовой части разработанного метода и возможных улучшений.

4. Проведен анализ влияния различных элементов реализации на результат работы и проведена оценка качества на данных с SemEval.

Список литературы

[1] *Plank, Barbara.* Multilingual Part-of-Speech Tagging with Bidirectional Long Short-Term Memory Models and Auxiliary Loss / Barbara Plank, Anders Søgaard, Yoav Goldberg // *CoRR.* — 2016. — Vol. abs/1604.05529. `http://arxiv.org/abs/1604.05529`.

[2] *Chen, Danqi.* A fast and accurate dependency parser using neural networks / Danqi Chen, Christopher Manning // Proceedings of the 2014 conference on empirical methods in natural language processing (EMNLP). — 2014. — Pp. 740–750.

[3] *Ma, Xuezhe.* End-to-end Sequence Labeling via Bi-directional LSTM-CNNs-CRF / Xuezhe Ma, Eduard Hovy // Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers). — Vol. 1. — 2016. — Pp. 1064–1074.

[4] Semeval-2015 task 3: Answer selection in community question answering / Preslav Nakov, Lluís Màrquez, Walid Magdy et al. // Proceedings of the 9th International Workshop on Semantic Evaluation (SemEval 2015). — 2015. — Pp. 269–281.

[5] SemEval-2016 Task 3: Community Question Answering / Preslav Nakov, Lluís Màrquez, Alessandro Moschitti et al. // Proceedings of the 10th International Workshop on Semantic Evaluation. — SemEval '16. — San Diego, California: Association for Computational Linguistics, 2016. — June.

[6] SemEval-2017 Task 3: Community Question Answering / Preslav Nakov, Doris Hoogeveen, Lluís Màrquez et al. // Proceedings of the 11th International Workshop on Semantic Evaluation. — SemEval '17. — Vancouver, Canada: Association for Computational Linguistics, 2017. — August.

[7] Distributed Representations of Words and Phrases and their Compositionality / Tomas Mikolov, Ilya Sutskever, Kai Chen et al. // Advances in Neural Information Processing Systems 26 / Ed. by C. J. C. Burges, L. Bottou, M. Welling et al. — Curran Associates, Inc., 2013. — Pp. 3111–3119. `http://papers.nips.cc/paper/`

5021-distributed-representations-of-words-and-phrases-and-their-compositionality pdf.

[8] Bag of Tricks for Efficient Text Classification / Armand Joulin, Edouard Grave, Piotr Bojanowski, Tomas Mikolov.

[9] *Pennington, Jeffrey.* Glove: Global vectors for word representation / Jeffrey Pennington, Richard Socher, Christopher Manning // Proceedings of the 2014 conference on empirical methods in natural language processing (EMNLP). — 2014. — Pp. 1532–1543.

[10] *Hochreiter, Sepp.* Long short-term memory / Sepp Hochreiter, Jürgen Schmidhuber // *Neural computation.* — 1997. — Vol. 9, no. 8. — Pp. 1735–1780.

[11] *Chiu, Jason PC.* Named Entity Recognition with Bidirectional LSTM-CNNs / Jason PC Chiu, Eric Nichols.

[12] *Filice, Simone.* KeLP at SemEval-2017 task 3: Learning pairwise patterns in community question answering / Simone Filice, Giovanni Da San Martino, Alessandro Moschitti // Proceedings of the 11th International Workshop on Semantic Evaluation (SemEval-2017). — 2017. — Pp. 326–333.

[13] *Moschitti, Alessandro.* Efficient convolution kernels for dependency and constituent syntactic trees / Alessandro Moschitti // European Conference on Machine Learning / Springer. — 2006. — Pp. 318–329.

[14] Selecting sentences versus selecting tree constituents for automatic question ranking / Alberto Barrón-Cedeno, Giovanni Da San Martino, Salvatore Romeo, Alessandro Moschitti // Proceedings of COLING 2016, the 26th International Conference on Computational Linguistics: Technical Papers. — 2016. — Pp. 2515–2525.

[15] Beihang-MSRA at SemEval-2017 Task 3: A Ranking System with Neural Matching Features for Community Question Answering / Wenzheng Feng, Yu Wu, Wei Wu et al. // Proceedings of the 11th International Workshop on Semantic Evaluation (SemEval-2017). — 2017. — Pp. 280–286.

[16] *Schlkopf, B.* Fast kernels for string and tree matching / B Schlkopf, K Tsuda, J Vert // *Advances in Neural Information Processing Systems.* — 2003. — Vol. 15, no. 1. — P. 296.

[17] *Collins, Michael.* New ranking algorithms for parsing and tagging: Kernels over discrete structures, and the voted perceptron / Michael Collins, Nigel Duffy // Proceedings of the 40th annual meeting on association for computational linguistics / Association for Computational Linguistics. — 2002. — Pp. 263–270.

[18] *Jeon, Jiwoon.* Finding similar questions in large question and answer archives / Jiwoon Jeon, W Bruce Croft, Joon Ho Lee // Proceedings of the 14th ACM international conference on Information and knowledge management / ACM. — 2005. — Pp. 84–90.

[19] FuRongWang at SemEval-2017 task 3: Deep neural networks for selecting relevant answers in community question answering / Sheng Zhang, Jiajun Cheng, Hui Wang et al. // Proceedings of the 11th International Workshop on Semantic Evaluation (SemEval-2017). — 2017. — Pp. 320–325.

[20] *Kingma, Diederik P.* Adam: A Method for Stochastic Optimization / Diederik P Kingma, Jimmy Lei Ba. — 2014.

[21] *Goodfellow, Ian.* Deep Learning / Ian Goodfellow, Yoshua Bengio, Aaron Courville. — MIT Press, 2016. — `http://www.deeplearningbook.org`.

[22] *Bahdanau, Dzmitry.* Neural Machine Translation by Jointly Learning to Align and Translate / Dzmitry Bahdanau, KyungHyun Cho, Yoshua Bengio. — 2017.