УДК 004.912

Аспектно-ориентированный анализ тональности текста на естественном языке

Aspect based sentiment analysis of a text in the natural language

1. **Гапанюк Ю.Е. (Gapanyuk Yu.E.), доцент кафедры «Системы обработки информации и управления» МГТУ им. Н.Э. Баумана, gapyu@bmstu.ru**
2. **Леонтьев А.В. (Leontiev A.V.), аспирант кафедры «Системы обработки информации и управления» МГТУ им. Н.Э. Баумана, alekseyl@list.ru**
3. **Латкин И.И. (Latkin I.I.), аспирант кафедры «Системы обработки информации и управления» МГТУ им. Н.Э. Баумана, igor.latkin@outlook.com**
4. **Ожегов Григорий Андреевич (Ozhegov G.A.), аспирант кафедры «Системы обработки информации и управления» МГТУ им. Н.Э. Баумана, grigory@ozhegov.name**
5. **Опрышко Александр Владимирович (Opryshko A.V.), аспирант кафедры «Системы обработки информации и управления» МГТУ им. Н.Э. Баумана, alexopryshko@yandex.ru**
6. **Чернобровкин С.В. (Chernobrovkin S.V.), аспирант кафедры «Системы обработки информации и управления» МГТУ им. Н.Э. Баумана, sergey.chernobrovkin@inbox.ru**

1. Введение

Объем контента, сгенерированного пользователями Интернета постоянно растет, что приводит к необходимости автоматизированного анализа больших объемов текстовых данных на естественном языке. Одной из важных задач анализа текстов является определение его тональности: позитивного, негативного или нейтрального отношения автора текстового сообщения к описываемой сущности.

Примерами данных, к которым применяется анализ тональности, являются, например, отзывы покупателей о товарах или услугах. Рассмотрим два примера отзывов:

1) «Это отличные наушники, мне понравилось качество звука и материалов, однако батарея садится очень быстро».
2) «Ужасный фильм, несмотря на отличный актерский состав и солидный бюджет».

Эти примеры наглядно показывают, что в одном отзыве могут соседствовать как положительные, так и отрицательные оценки разных аспектов комментируемых сущностей.

Если проводить анализ тональности для полного текста отзыва, то результаты анализа могут быть неточными и даже противоречивыми. Что должно быть важнее для алгоритма определения полярности: «ужасный фильм» или «отличный актерский состав»? Или в целом такой отзыв нужно считать нейтральным? Наверное, наиболее правильным решением в этом случае будет решение о том, что попытка оценки полярности для всего предложения заведомо некорректна. В такой ситуации более корректным решением будет независимый анализ полярности для отдельных аспектов отзыва. Задачу анализа тональности для отдельных аспектов текста в соответствии с [1] принято называть задачей аспектного анализа тональности текста (ABSA – Aspect-Based Sentiment Analysis).

В статье мы рассмотрим способ решения данной задачи, основанный на использовании комбинации методов машинного обучения.

2. Постановка задачи

Если используется классический алгоритм анализа тональности для текста T, то на выходе алгоритма определения тональности для каждого текста T формируется пара (T, P), где P – polarity (полярность, характеристика отношения к заданному тексту).

В классическом способе анализа тональности рассматриваются три варианта полярности: положительная, отрицательная и нейтральная. С точки зрения машинного обучения такая задача является задачей многоклассовой (в данном случае трехклассовой) классификации. Используемый классификатор должен отнести текст к одному из трех классов. Иногда нейтральный класс не рассматривается, тогда с точки зрения машинного обучения задача упрощается до задачи бинарной (двухклассовой) классификации.

Если мы рассматриваем аспектный анализ тональности текста T, то результатом работы алгоритма определения тональности является множество троек вида (E, A, P), где E – entity (комментируемая сущность), A– aspect (аспект сущности), P – polarity (полярность, характеристика отношения к конкретному аспекту сущности). При этом параметры E и A должны быть выделены из текста T.

Как и в классическом способе анализа тональности, в аспектном анализе тональности рассматриваются положительная, отрицательная и нейтральная полярности, возможна постановка задачи с исключением нейтральной полярности.

Выделим тройки (E, A, P) из рассмотренных выше примеров отзывов. Далее приведены такие тройки в виде E#A: P.

1) Наушники#Общее: положительно
   Наушники#Звук: положительно
   Наушники#Материал: положительно
   Наушники#Батарея: отрицательно
2) Фильм#Общее: отрицательно
   Фильм#Актеры: положительно
   Фильм#Бюджет: положительно

Рассмотренный пример показывает, что задача аспектного анализа тональности текста намного сложнее задачи классического анализа тональности. Для ее решения требуется использовать комбинацию из нескольких методов машинного обучения.

В соответствии с [1] задача аспектного анализа тональности текста состоит из трех подзадач. Подзадачи выполняются последовательно и могут рассматриваться как три шага алгоритма аспектного анализа тональности текста:

1) Определение категории аспекта (ACD, aspect category detection).
   В результате решения подзадачи формируется множество пар (E, A) для заданного текста T.
2) Выделение выражения цели высказывания (OTE, opinion target expression).
   В результате решения подзадачи выделяются фрагменты предложений, относящиеся к каждому аспекту.
3) Определение полярности (PD, polarity detection).
   На основе выделенных на предыдущем шаге фрагментов предложений для каждого аспекта определяется его полярность: положительная, отрицательная или нейтральная.

Таким образом, для выполнения аспектного анализа тональности текста необходимо последовательно выполнить три рассмотренных шага (решить три подзадачи). Далее в данной статье мы рассматриваем варианты решения каждой из подзадач с помощью различных методов машинного обучения.

Статья организована следующим образом. Поскольку для любого метода машинного обучения очень важным является выделение подходящих признаков, то в

разделе 3 рассматриваются варианты выделения признаков для оценки текстов. Разделы 4, 5, 6 посвящены вариантам решения подзадач 1, 2, 3. В разделе 7 рассмотрена структура системы аспектно-ориентированного анализа тональности текста на основе метаграфового подхода. В разделе 8 приведены результаты экспериментов.

3. Выделение признаков

В данном разделе кратко рассмотрим существующие варианты выделения признаков из текста, наиболее важным из которых является векторное представление слов. Преобразование текстовой информации в числовое (особенно в векторное) представление позволяет использовать стандартные модели машинного обучения, которые созданы в основном для работы с векторными данными.

В настоящее время наиболее распространены частотные способы формирования векторного представления текстовой информации. Самый простой способ это представление исходных текстов в виде векторов следующего вида: $[i_1, i_2, \dots, i_m]$, где m – количество слов в тексте, i_k – количество вхождений k-го слова в тексте. Данный способ довольно распространён и прост в реализации, но имеет очевидные недостатки. В частности, при сравнении двух векторов используется точное совпадение слов, и мы не можем различить семантически схожие слова, такие как синонимы, гипонимы и гиперонимы.

Более сложным способом является взвешивание слов с использованием меры TF-IDF. Преимуществом этой меры является то, что каждое слово, которое наиболее специфично для текущего текста, будет иметь больший вес.

Еще более сложным способом является динамическое построение n-мерного числового пространства для заданного текста. Данный подход является достаточно новым, но в настоящее время используется все чаще. Основная задача заключается в нахождении числового вектора в n-мерном пространстве для каждого слова. Такое векторное представление основывается на идее контекстной близости: слова, встречающиеся в тексте рядом с одинаковыми словами (следовательно, имеющими схожий смысл), в векторном представлении будут иметь близкие координаты векторов-слов.

Можно выделить два наиболее популярных способа преобразования слов в векторное представление – Word2Vec [2] и GloVe [3]. Эти методы используют разный

подход к обучению моделей. Word2Vec является предиктивной моделью, в то время как модель GloVe основана на подсчете появлений слов в разных контекстах.

Word2Vec был разработан группой исследователей Google в 2013 году. В Word2Vec существуют два основных алгоритма обучения: CBOW (Continuous Bag of Words) и Skip-gram. CBOW («непрерывный мешок слов») – модельная архитектура, которая предсказывает текущее слово, исходя из окружающего его контекста. Архитектура типа Skip-gram действует иначе: она использует текущее слово, чтобы предугадывать окружающие его слова. Пользователь Word2Vec имеет возможность выбирать один из двух алгоритмов при обучении модели. Порядок слов контекста не оказывает влияния на результат ни в одном из этих алгоритмов. Получаемые на выходе координатные представления векторов-слов позволяют вычислять «семантическое расстояние» между словами. Именно основываясь на контекстной близости этих слов, технология Word2Vec совершает свои предсказания. Так как инструмент Word2Vec основан на обучении нейронной сети, чтобы добиться его наиболее эффективной работы, необходимо использовать большие корпусы для его обучения. Это позволяет повысить качество предсказаний.

В моделях, основанных на подсчете появлений слов в разных контекстах (GloVe), обучение векторов происходит преимущественно уменьшением размерности матрицы совместного появления. Первоначально строится матрица большого размера (количество слов на размерность контекста) совместного появления слов, т.е. для каждой строки (слова) подсчитывается, сколько раз данное слово встречалось в том или ином контексте. Далее данная матрица факторизуется, превращаясь в матрицу меньшего размера (количество слов на количество признаков).

Для представления в виде вектора всего документа (а не отдельных слов) можно использовать различные эвристики по усреднению векторов слов (например, брать среднее значение или взвешивать на основе меры TF-IDF). Помимо эвристических методов существует алгоритм усреднения Doc2Vec, основанный на алгоритме Word2Vec.

4. Определение категории аспекта

Задача определения категорий аспектов, представленных в предложении – это задача многометочной классификации (multi-label classification).

Если в задаче многоклассовой классификации алгоритм классификации должен отнести объект к одному элементу в заданном множестве классов, то в задаче многометочной классификации алгоритм классификации должен отнести объект к подмножеству заданного множества классов, то есть сопоставить классифицируемому объекту несколько классов (которые в этом случае принято называть метками). Таким образом, алгоритм многоклассовой классификация – это частный случай алгоритма многометочной классификации с единичной мощностью искомого подмножества классов.

В данном случае множеством целевых классов является множество пар (E, A), которые должен распознать алгоритм многометочной классификации в исходном тексте.

Существуют два основных подхода к решению задачи многометочной классификации:

1) переход к множеству бинарных классификаторов, где каждый классификатор отвечает за один класс (схема аналогичная подходу one-vs-all [4]);
2) использование классификаторов, изначально поддерживающих многометочную классификацию, например:
   - алгоритм «Multi label k-nearest neighbors» [5];
   - алгоритм «Back propagation Multi Label Learning» [6].

На практике часто хорошие результаты показывает множество SVM-классификаторов, обученных в режиме one-vs-all [7].

В качестве признаков для классификаторов целесообразно использовать средние, минимальные и максимальные значения точности, полноты и F1-меры, вычисленные для каждого слова, стеммы слова и POS-тегов, а также векторные представления слов [8].

На вход алгоритма определения категории аспекта подается текст предложения, на выходе алгоритма формируется множество категорий аспектов - пар (E, A), найденных в этом предложении с помощью одного из алгоритмов многометочной классификации.

5. Выделение выражения цели высказывания

Выделение выражения цели высказывания (OTE) – это задача последовательной разметки (sequence labeling) текста. Для её решения можно было бы классифицировать

каждое слово в отдельности. Однако на практике лучше себя показывают подходы, учитывающие также предыдущие и следующие слова в предложении.

Классическим примером задачи последовательной разметки является разметка частей речи для слов в предложении. Например, рассмотрим предложение «Мама мыла раму». Слово «мыла» может быть классифицировано и как существительное среднего рода единственного числа в родительном падеже (мыло), и как глагол прошедшего времени (мыть). Однако используя контекст (окно в одно или несколько слов назад и вперед) становится понятно, что три существительных, расположенных подряд – это менее вероятный исход, чем глагол, связывающий два существительных.

В задаче определения OTE в качестве меток используется разметка IOB:

1) B (begin) – начало OTE;
2) I (inside) – продолжение OTE;
3) O (outside) – слово не входит в OTE.

Рассмотрим на примере: «Ужасный фильм, несмотря на отличный актерский состав». В этом предложении два OTE: «фильм» и «актерский состав».

Результатом работы классификатора должен быть следующий вектор:

$$(O,B,O,O,O,B,I)$$

Последовательная разметка текста – это разновидность задачи структурной классификации. Структурная классификация отличается от обычной тем, что для предсказания очередной метки в последовательности наряду с исходными признаками также учитывается результат классификации соседних элементов последовательности.

Одним из методов решения этой проблемы является алгоритм Conditional Random Field [9]. Conditional Random Field (CRF) – это вероятностная графовая модель, которая учитывает не только унарную вероятность отнесения элемента (в нашем случае – слова) к определенному классу, но и парную вероятность нахождения двух классифицированных элементов рядом.

В качестве признаков для классификации целесообразно использовать следующие:

1) Морфологические (наличие заглавных букв, цифр и пунктуации);
2) Лексические:
    - POS-теги (метки частей речи);
    - префиксы и суффиксы слова;
    - аспектные выражения.

3) Векторное представление слов (в частности, с использованием подхода Word2Vec).

Кроме признаков текущего слова алгоритм должен учитывать признаки соседних слов в предложении.

Таким образом, на данном шаге вначале формируются IOB-вектора. Если наложить IOB-вектор как маску на исходное предложение, то мы получим множество слов исходного предложения, относящихся к конкретной категории аспекта – паре (E, A).

6. Определение полярности выражения

6.1. Выделение признаков

Существующие подходы можно, условно, разделить на две группы по способу выделения признаков:

1. подготовка большого числа признаков вручную;
2. преобразование текстового представления информации в числовое (Word2Vec, GloVe).

Создаваемые вручную признаки в соответствии с [8] можно разделить на следующие категории:

1. морфологические:
   - частотные (например, число восклицательных и вопросительных знаков);
   - булевые (наличие того или иного знака препинания в конце предложения).
2. на основе информации о частях речи (число существительных, прилагательных и т.д.);
3. на основе пар E#A (количество аспектов и сущностей);
4. на основе лексиконов тональности.

6.2. Переход от аспектной постановки задачи к классической

Классическая задача определения тональности текста может быть сформулирована как задача классификации, где на входе алгоритма – множество признаков текста (*TF*), а на выходе класс тональности: положительный, нейтральный или отрицательный – $f : TF \mapsto класс$.

То есть в классической постановке задачи одному тексту соответствует один класс тональности.

В аспектно-ориентированном подходе задача усложняется тем, что на входе теперь не только текст, но также множество категорий аспектов (*AC*) и множество выражений цели высказывания (*OTE*). На выходе также множество классов – по одному на каждый аспект – $f:\left(TF,\{AC\},\{OTE\}\right)\mapsto\{класс\}$.

Чтобы перейти от аспектной постановки задачи к классической необходимо предложить алгоритм соответствия (маппинга) аспектов в предложения. Такой алгоритм на вход принимает текстовые признаки, категории аспектов и выражения цели высказывания, а на выходе мы имеем множество преобразованных текстовых признаков (*TTF*) – $map:\left(TF,\{AC\},\{OTE\}\right)\mapsto\{TTF\}$.

После такого преобразования каждому набору преобразованных признаков будет соответствовать уже только один класс. И задача будет сведена к классической – $f:TTF\mapsto класс$.

Рассмотрим 2 возможных метода маппинга аспектов на текст:

1) Разделение предложений на части по факту принадлежности к аспекту.
2) Изменение веса векторов слов.

6.2.1. Разделение предложений на части по факту принадлежности к аспекту

Для каждого *OTE* находим расстояние от него до остальных слов в тексте. Если расстояние превышает некий порог, то отсекаем эти слова. В результате получаем часть предложения, которая относится к текущему аспекту.

В качестве метрики расстояния между словами тексте в данном случае целесообразно использовать:

1) линейное расстояние (количество слов в порядке их следования в тексте);
2) древовидное расстояние (наименьший путь между двумя словами в синтаксическом дереве, построенном для данного предложения).

Рассмотрим на примере: «Звук хороший, но управление неудобное». Здесь два ОТЕ: «звук» и «управление». Возьмём пороговое расстояние равное двум, получим две части предложения:

1) «звук хороший, но»;

2) «но управление неудобное».

6.2.2. Изменение веса векторов слов

Этот метод был предложен в статье [10]. Алгоритм авторов выглядит следующим образом:

1) Строим векторное представление предложения.
2) Получаем вероятность принадлежности каждого слова к аспекту.
3) Выбираем слова, имеющие вероятность отнесения к аспекту больше пороговой (получаем множество *OTE*).
4) Находим древесное расстояние от каждого *OTE* до остальных слов.
5) Умножаем вероятность отнесения слова к аспекту на расстояние до текущего *OTE*.
6) В результате получаем вектор коэффициентов усиления – по одному на каждое слова для текущего аспекта.
7) Умножаем исходное векторное представление предложения на усиливающий вектор, тем самым слова, относящиеся к текущему аспекту, будут иметь значительно большую норму в векторном представлении. Этот факт затем учитывается на этапе определения тональности, благодаря использованию сверточной нейронной сети со слоем типа *max_pool*.

6.3. Определение полярности

В результате выполнения предыдущих шагов мы получили документы, представленные в виде векторов. Теперь задачу определения полярности можно решать как обычную задачу многоклассовой (или бинарной, в случае отсутствия нейтрального класса) классификации.

Для решения этой задачи наиболее часто используют следующие варианты классификаторов:

1) наивный байесовский классификатор;
2) различные линейные модели, в том числе SVM (Support Vector Machine) – метод опорных векторов;
3) деревья решений;
4) нейросетевые алгоритмы (feed-forward, рекуррентные, рекурсивные или сверточные сети).

Результаты экспериментов с различными классификаторами приведены в разделе 8.

7. Структура системы аспектно-ориентированного анализа тональности текста на основе метаграфового подхода

С точки зрения подхода гибридных интеллектуальных информационных систем (ГИИС), предложенного в [11], задача аспектно-ориентированного анализа тональности текста может быть рассмотрена как задача, относящаяся к модулю подсознания (МП) ГИИС и решаемая методами машинного обучения. Структура предлагаемой системы представлена на рис. 1.

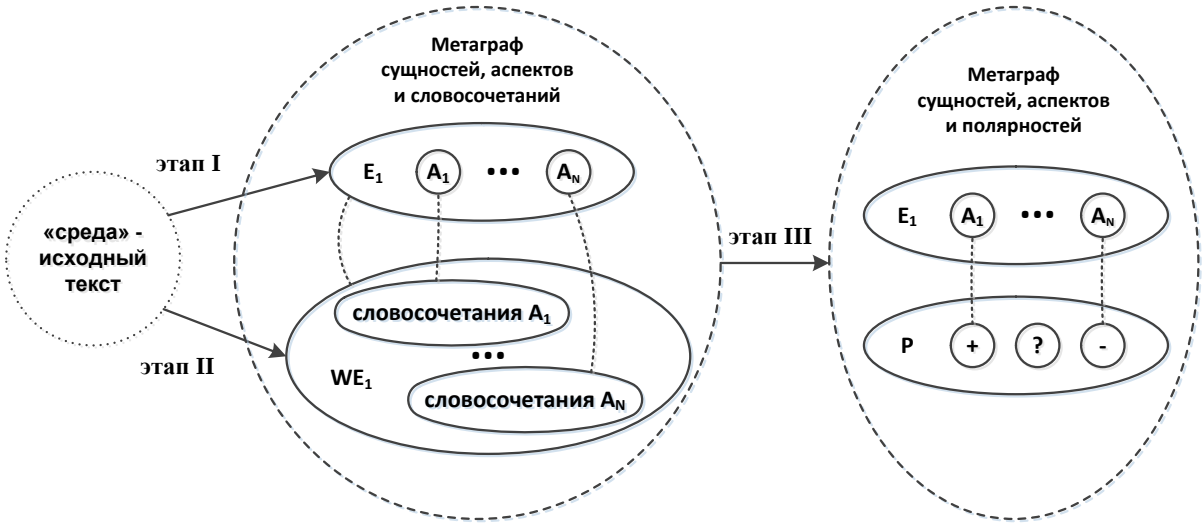

Рис. 1. Структура системы аспектно-ориентированного анализа тональности текста на основе метаграфового подхода.

В качестве «среды» здесь выступает исходный текст.

На этапе I происходит определение категории аспекта (ACD, aspect category detection). В результате формируется метаграф, содержащий метавершины сущностей (на рис. 1 представлена одна метавершина E_1) и вложенные в нее аспекты A_1 … A_N. Подходы к выполнению этапа I рассмотрены в разделе 4.

На этапе II решается задача определения цели высказывания (OTE, opinion target expression). В результате из исходного текста выделяются словосочетания, относящиеся к аспектам сущностей. На рис. 1 представлена метавершина WE_1, содержащая словосочетания. Между метавершинами E_1 и WE_1 и вложенными

вершинами устанавливаются соответствующие связи. Подходы к выполнению этапа II рассмотрены в разделе 5.

На этапе III решается задача определения полярности (PD, polarity detection). В результате каждому аспекту сущности приписывается соответствующая полярность (оценка пользователем конкретного аспекта). Полярности показаны в виде отдельной метавершины P, включающей три вершины: положительная (+), отрицательная (-) и нейтральная (?).Подходы к выполнению этапа III рассмотрены в разделе 6.

Особенность задачи состоит в том, что в получившемся метаграфе в основном устанавливаются связи между вершинами различных метавершин. Во многих метавершинах вложенные вершины не связаны между собой (в этом нет необходимости для решения задачи), поэтому такие метавершины в большей степени напоминают гиперребра гиперграфа.

Сформированный метаграф может быть использован на следующих этапах обработки текста.

8. Эксперименты

8.1. Наборы данных

В качестве наборов данных для обучения использовался датасет с отзывами покупателей ноутбуков, предоставленный в соревновании SemEval 2016 [12].

В этом датасете 2500 предложений, и каждому соответствует несколько мнений– троек (E, A, P).

Пример элемента датасета:

```
<sentence id="B00KMRGF28_112_AG0H7UG6OX2GU:5">
  <text>i am giviing it four stars instead of five because of no disk drive and the crashing</text>
  <Opinions>
    <Opinion category="LAPTOP#GENERAL" polarity="positive"/>
    <Opinion category="LAPTOP#DESIGN_FEATURES" polarity="negative"/>
    <Opinion category="LAPTOP#OPERATION_PERFORMANCE" polarity="negative"/>
  </Opinions>
</sentence>
```

8.2. Выделение признаков

Для выделения признаков из текста предложений была использована программная реализация модели Word2Vec от Google [13]. В ней каждому слову соответствует 300-мерный числовой вектор.

Алгоритм выделения признаков, использованный в рамках эксперимента:

1. Разбить предложение на слова.
2. Каждое слово заменить на его векторное представление, если это слово присутствует в модели и его нет в списке стоп-слов.
3. Усреднить вектора всех слов, чтобы получить один 300-мерный вектор.
4. Нормализовать вектор.

8.3. Описание эксперимента

Задача: подобрать оптимальный классификатор, его параметры и порог его срабатывания.

Алгоритм подбора:

1. Для каждого типа классификатора:
    1.1. Для каждого значения оптимизируемого параметра:
        1.1.1. Обучить классификатор.
        1.1.2. Для каждого порога срабатывания классификатора:
            1.1.2.1. Вычислить среднюю F1-меру на трех валидационных выборках.
        1.1.3. Выбрать максимальное значение F1 и соответствующий ему порог среди полученных значений.
    1.2. Выбрать значение параметра с максимальным значением F1.
2. Выбрать классификатор с максимальным значением F1.

В сравнении участвовали следующие алгоритмы:

1) SVM (с rbf-ядром) (оптимизировался параметр C, позволяющий регулировать отношение между максимизацией ширины разделяющей полосы и минимизацией суммарной ошибки).
2) Random Forest (оптимизировалось число классификаторов).
3) Gaussian Naive Bayes (модель не имеет гиперпараметров).
4) Multi Layer Perceptron с 20 нейронами в скрытом слое (оптимизировалось число итераций обучения).

5) Multi Layer Perceptron с 156 нейронами в скрытом слое (оптимизировалось число итераций обучения).

8.4. Методика оценки качества классификации

Для оценки качества классификации в экспериментах используется мера F1, так как, в отличие от меры точности (accuracy score), F1-мера учитывает несбалансированность размеров выборок, относящихся к разным классам.

$$F1 = 2 * \frac{precision * recall}{precision + recall},\ precision = \frac{TP}{TP + FP},\ recall = \frac{TP}{FN + TP}$$

TP – True Positives. Количество правильно определенных алгоритмом положительных примеров. То есть количество предложений, которые реально относятся к тому аспекту, за который отвечает данный классификатор, и классификатор это определил.

FP – False Positives. Количество примеров, неправильно классифицированных алгоритмом, как положительные. То есть количество примеров, которые реально не относятся к тому аспекту, за который отвечает данный классификатор, но классификатор посчитал их относящимися к данному аспекту.

FN – False Negatives. Количество примеров, неправильно определенных алгоритмом, как отрицательные. То есть количество предложений, которые реально относятся к тому аспекту, за который отвечает данный классификатор, но классификатор этого определить не смог.

В связи с необходимостью учитывать специфику задачи многометочной классификации, алгоритм подсчета F1-меры выглядит так:

1. Для каждого предсказания, представленного в виде N-мерного вектора, где N – количество меток классификации:
   1.1. Получить реальное множество меток – *actuals.*
   1.2. Для каждого элемента вектора предсказания:
      1.2.1. Если предсказанная вероятность принадлежности к текущему классу больше пороговой, добавить текущую метку в множество предсказанных – *predicted.*
      1.2.2. Изменить значения TP, FP и FN следующим образом: $TP = TP + \left|actuals \cap predicted\right|$, $FP = FP + \left|predicted - actuals\right|$, $FN = FN + \left|actuals - predicted\right|$.

2. Вычислить *precision*, *recall* и *F1*, используя аккумулированные значения *TP*, *FP* и *FN*.

8.5. Результаты

На рис. 2-5 представлены кривые обучения для предложенных ранее алгоритмов.

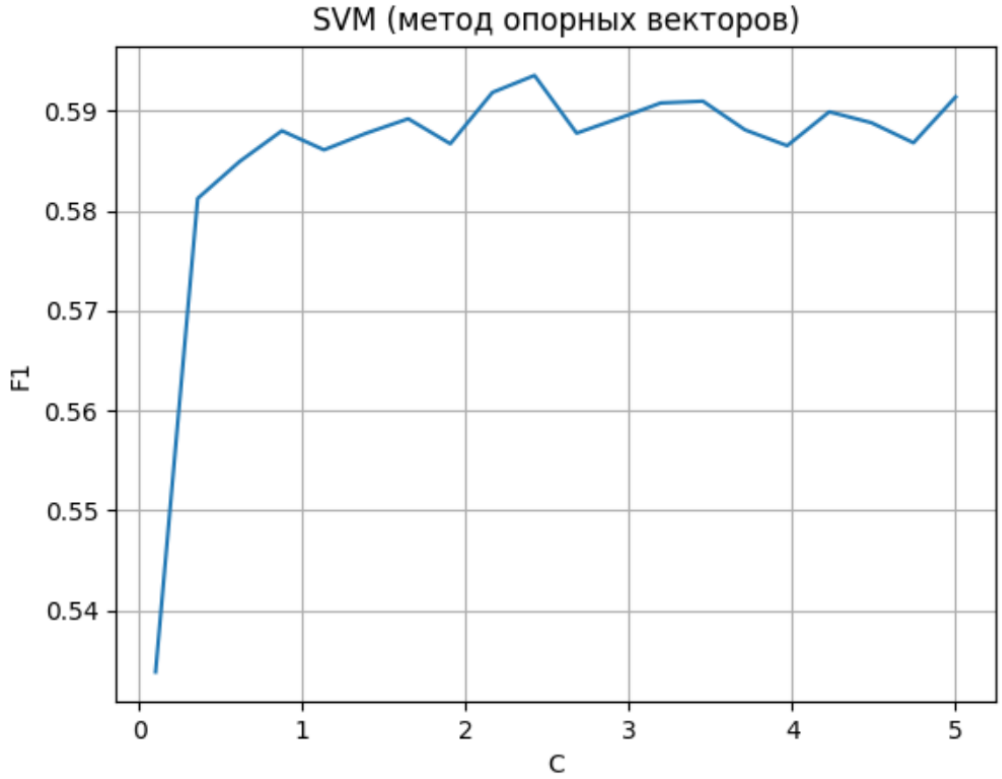

Рис. 2. Кривая обучения для SVM.

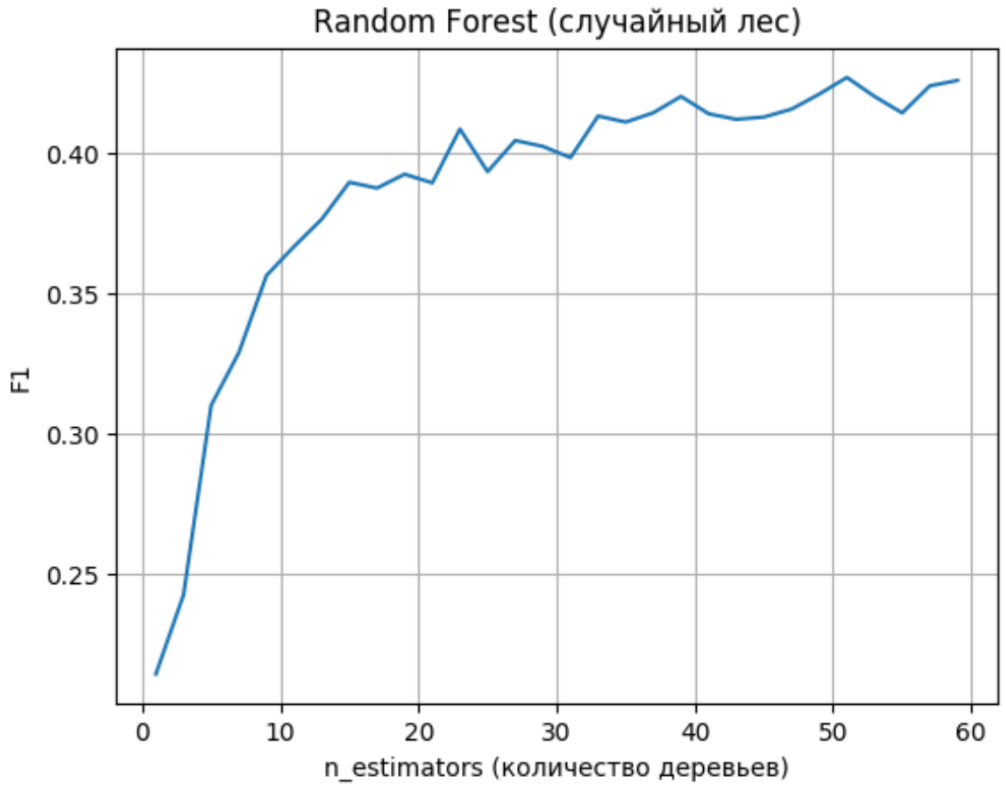

Рис. 3. Кривая обучения для Random Forest.

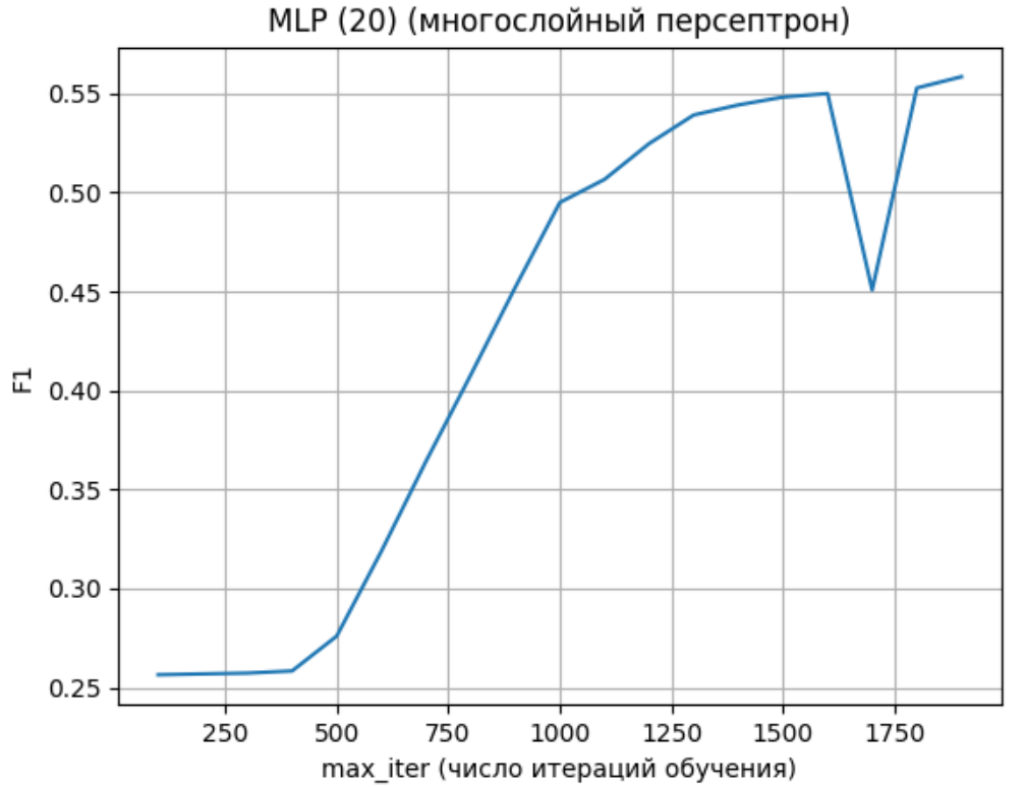

Рис. 4. Кривая обучения для Multi Layer Perceptron с 20 нейронами в скрытом слое.

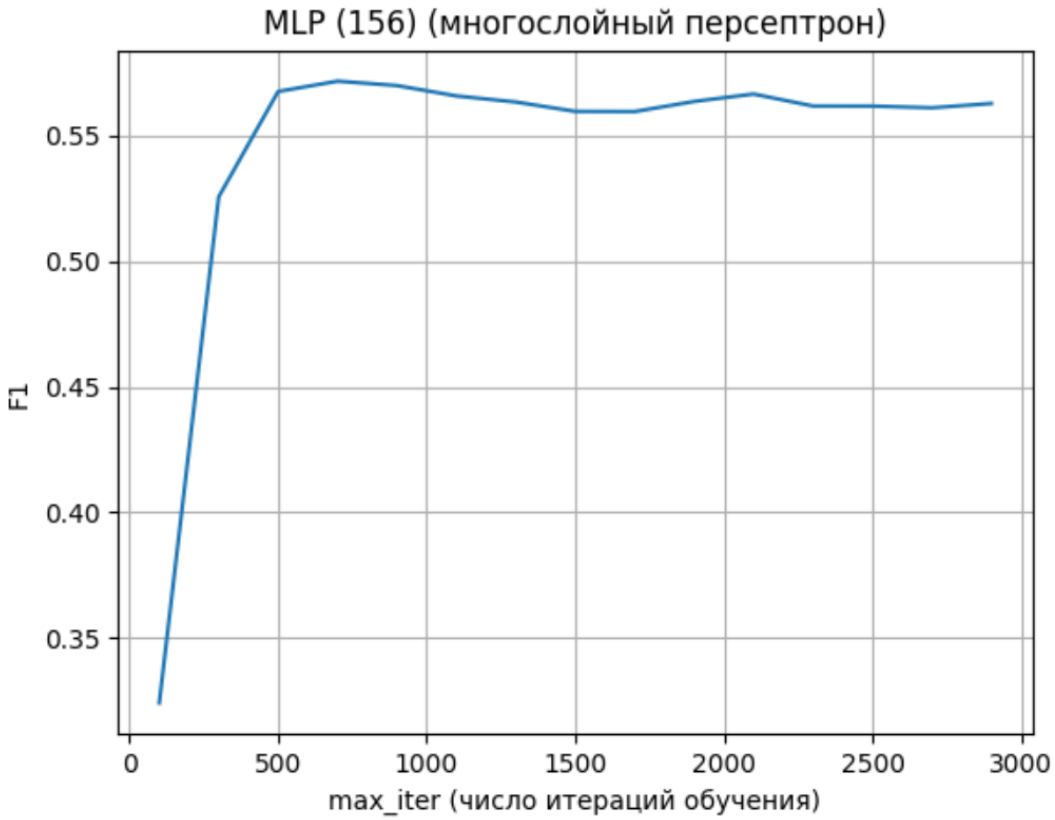

Рис. 5. Кривая обучения для Multi Layer Perceptron со 156 нейронами в скрытом слое.

Результаты сравнения классификаторов представлены в табл. 1.

Таблица 1. Результаты сравнения классификаторов

Классификатор	**Оптимальное значение параметра**	**Максимальное среднее значение F1-меры**
SVM	C = 3.1947368421052635	**0.59228**
Random Forest	n_estimators = 55	0.42217
Gaussian Naive Bayes	-	0.48550
MLP (20)	max_iter = 1800	0.55624
MLP (156)	max_iter = 900	0.57279

Здесь у модели SVM оптимизировался параметр C, позволяющий регулировать отношение между максимизацией ширины разделяющей полосы и минимизацией суммарной ошибки. У Random Forest оптимизировалось число классификаторов - n_estimators. У Gaussian Naive Bayes никакие параметры не оптимизировались. У обоих

многослойных персептронов (MLP) оптимизировалось число итераций обучения - max_iter.

Из табл. 1 можно сделать вывод, что лучше всего с задачей справился SVM-классификатор с результатом F1 = 0.59228. Рассмотрим график подбора оптимального порога классификации для этого классификатора. Этот порог напрямую влияет на значения TP, FP и FN, используемые при подсчете F1-меры.

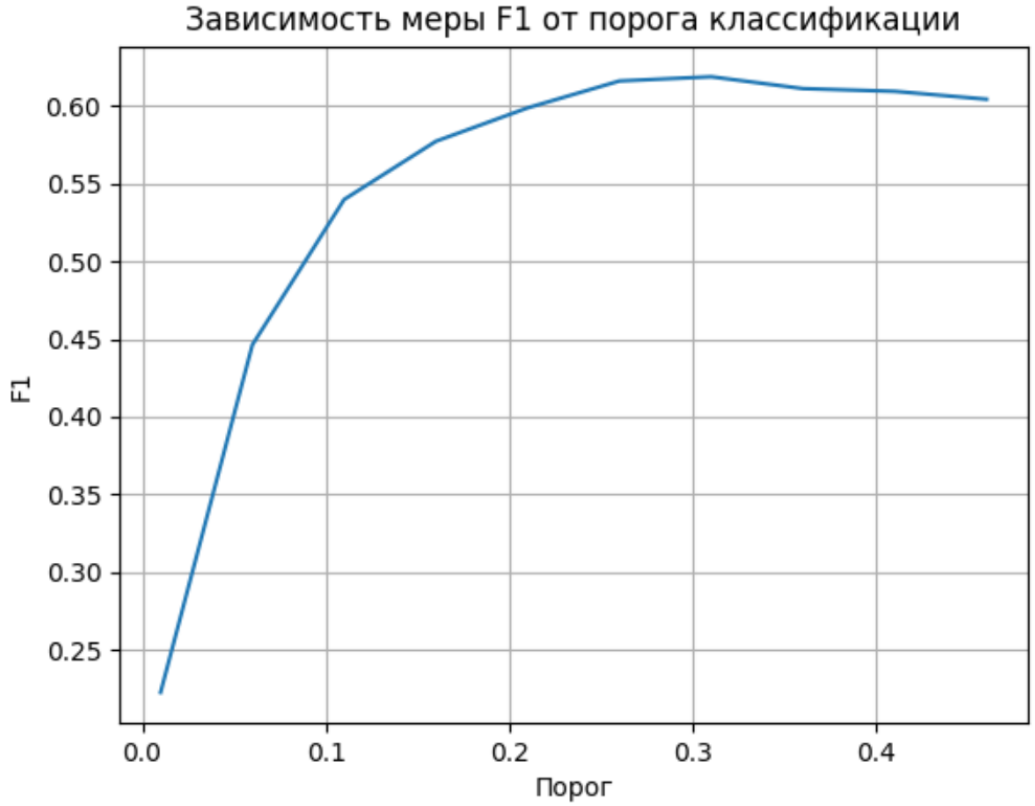

Рис. 6. График зависимости F1-меры от порога классификации.

Из графика на рис. 6 видно, что максимуму F1-меры (0.61) соответствует порог равный 0.31.

9. Выводы

Аспектно-ориентированный анализ тональности текстов – достаточно обширное и наукоемкое направление исследований в области обработки текстов на естественном языке.

В данной статье задача аспектно-ориентированного анализа тональности текстов представлена как композиция различных методов машинного обучения. Прежде всего, это методы классификации, в частности многометочной и многоклассовой, а также бинарной.

По результатам экспериментов, представленных в данной статье видно, что для задачи определения категорий аспектов на данном наборе данных лучше всего себя показывает SVM-классификатор. Максимальное достигнутое в ходе экспериментов значение F1-меры равно ~0.59.

Список литературы

1. John Pavlopoulos, Ion Androutsopoulos. Aspect Term Extraction for Sentiment Analysis: New Datasets, New Evaluation Measures and an Improved Unsupervised Method. Proceedings of the 5th Workshop on Language Analysis for Social Media (LASM), 2014, pp. 44–52.

2. T. Mikolov, I. Sutskever, K. Chen, G.S. Corrado, J. Dean. Distributed representations of words and phrases and their compositionality. Advances in neural information processing systems, 2013, pp. 3111-3119.

3. Jeffrey Pennington, Richard Socher and Christopher D. Manning. GloVe: Global Vectors for Word Representation. Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP), 2014, pp. 1532–1543.

4. Grigorios Tsoumakas, Ioannis Katakis. Multi-label classification: an overview. International Journal of Data Warehousing & Mining 3 (3) 2007, pp. 1-13.

5. M.L. Zhang, Z.H. Zhou. ML-KNN: A lazy learning approach to multi-label learning. Pattern Recognition. 40 (7) 2007, pp. 2038–2048.

6. M.L. Zhang, Z.H. Zhou. Multi-label neural networks with applications to functional genomics and text categorization. IEEE Transactions on Knowledge and Data Engineering, 2006. pp. 1338–1351.

7. Dionysios Xenos, Panagiotis Theodorakakos, John Pavlopoulos, Prodromos Malakasiotis and Ion Androutsopoulos. Ensembles of Classifiers and Embeddings for Aspect Based Sentiment Analysis. Proceedings of the 10th International Workshop on Semantic Evaluation (SemEval-2016), 2016, pp. 312-317.

8. Rafael Michael Karampatsis, John Pavlopoulos and Prodromos Malakasiotis. AUEB: Two Stage Sentiment Analysis of Social Network Messages. Proceedings of the 8th International Workshop on Semantic Evaluation (SemEval 2014), 2014, pp. 114–118.

9. Wallach, H.M.: Conditional random fields: An introduction. Technical report MS-CIS-04-21, University of Pennsylvania, 2004.

10. Wang Bo and Min Liu. Deep Learning for Aspect-Based Sentiment Analysis. Technical report, Stanford University, 2015.

11. Черненький В.М., Терехов В.И., Гапанюк Ю.Е. Структура гибридной интеллектуальной информационной системы на основе метаграфов. Нейрокомпьютеры: разработка, применение. 2016. Выпуск №9. С. 3-14.

12. SemEval-2016 Task 5: Aspect-Based Sentiment Analysis. Режим доступа: http://alt.qcri.org/semeval2016/task5/ (дата обращения 03.04.2017).

13. Word2Vec. Tool for computing continuous distributed representations of words. Режим доступа: https://code.google.com/archive/p/Word2Vec/ (дата обращения 03.04.2017).