УДК 004.056

Студ. А.С. Демещик
Науч. рук. канд. техн. наук А.С. Кобайло
(кафедра программной инженерии, БГТУ)

КЛАССИФИКАЦИЯ ОБЪЕКТОВ С ПОМОЩЬЮ ИСКУССТВЕННОГО ИНТЕЛЛЕКТА

Задача классификации объектов является формализованной задачей, т.е. имеет четкие постановку и алгоритм решения, в которой имеется множество объектов, которые относятся к определенным классам. Существует подмножество этого множества, в котором классы объектов точно определены. Это подмножество называется выборкой. Принадлежность к классам остальных объектов неизвестна. Суть задачи классификации: указать номер(наименование) класса, к которому принадлежит объект из исходного множества с помощью алгоритма, который и занимается классификацией.

Рассмотрим задачи классификации чуть подробнее. В математической статистике данные задачи называются также задачами дискриминантного анализа. Можно выделить несколько типологий, по которым можно классифицировать данные задачи:

- Типы классов:
  - Двухклассовая классификация. Этот случай, довольно часто используется при решении более сложных задач.
  - Многоклассовая классификация.
  - Непересекающиеся классы.
  - Пересекающиеся классы. В этом случае объект может относиться сразу к нескольким классам.
  - Нечеткие классы. В задачах с таким типом классов требуется определить принадлежность объекта к каждому классу, обычно это число от 0 до 1.
- Типы входных данных:
  - Признаковое описание. Каждый объект представлен набором своих характеристик, называемых признаками. Признаки могут быть числовыми и нечисловыми.
  - Матрица расстояний между объектами. В этом случае, каждый объект представлен расстояниями до всех остальных объектов выборки.
  - Временной ряд или сигнал. Объект являет собой последовательность измерений во времени.
  - Изображение или видеоряд.

- Существуют более сложные типы, например, графы, тексты и т.д., которые, чаще всего, приводятся к первому либо второму типу путем предварительной обработки.

Задачу классификации можно выразить с помощью математики следующим образом:

X – множество объектов, Y – множество номеров (наименований) классов. Существует неизвестное целевая зависимость: $y^*: X \to Y$, значения которой известны лишь на объектах конечной выборки:

$$X^m = \{(x_1,y_1),\ldots,(x_m,y_m)\}$$

Требуется построить алгоритм $a: X \to Y$, способный классифицировать произвольный объект $x \in X$.

Существует несколько подходов к классификации объектов:

- Классификация вручную. Наиболее простой подход, который идеально подходит для малого множества объектов, однако становится неприемлем, когда требуется обработка большого количества данных за короткий срок.
- Написание правил для классификации. Этот подход намного лучше первого, т.к. позволяет автоматизировать процесс. Плюс построение правил вручную позволяет дать лучшую точность в сравнении с машинным обучением. Однако создание и изменение правил требует постоянного контроля.
- Машинное обучение. В данном случае, набор правил, необходимых для классификации объектов, вычисляется из обучающих данных. Обучающие данные – некоторое количество правильно размеченных данных из каждого класса. В этом подходе сохраняется необходимость ручной классификации, но она значительно проще написания правил. Плюс, классификация может происходить параллельно с работой системы.

Из вышесказанного можно сделать вывод, что машинное обучение, в решении задачи классификации, является наиболее приемлемым вариантом. Данные задачи решаются, в частности, при помощи искусственных нейронных сетей при постановке эксперимента как в виде обучения с учителем, так и в виде обучения без учителя.

Применение задачи классификации

При выполнении научной работы передо мной стоит задача создания веб-приложения с использованием ReactJS, которое будет работать с сервером, написанном на языке программирования C#. Который, в свою очередь, будет взаимодействовать с нейронной сетью, реализованной с помощью Keras фреймворка. Данная нейронная сеть решает задачу классификации текстов на основе признаков. Можно

сказать, что решаемая задача является многоклассовой и задачей с признаковым описанием.

Решение данной задачи происходило следующими этапами:

- сбор данных, которые будут являться конечной обучающей выборкой для сети;
- предобработка данных: удаление редких/частотных слов; стемминг/лемматизация;
- построение и обучение нейронной сети;
- оценка результатов обучения.

В качестве нейронной сети использовалась сеть LSTM – long-short-termmemory – сеть долгой краткосрочной памяти.

LSTM сеть – это рекуррентная сеть. Это значит, что, в отличие от обычных нейронных сетей, сеть умеет сохранять информацию (рис. 1), что позволяет использовать предыдущие выводы для новых решений.

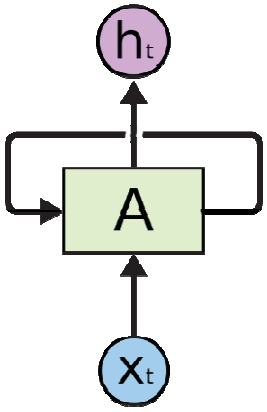

Рисунок 1 – Схема рекуррентной сети

Данный тип сетей решает задачу по запоминанию ранее полученной информации для использования в текущей. И рекуррентная сеть идеально справляется, когда расстояние между текущей задачей и необходимым данными невелико. Но когда оно увеличивается, то данные сети уже неспособны на воспроизведение прошлых результатов.

Чтобы решить эту проблемы были разработаны LSTM сети (рис. 2).

Чтобы пояснить наиважнейшее отличие от других рекуррентных сетей, я приведу схему обычной RNN сети (рис. 3).

Как можно заметить, сравнив 2 рисунка, в LSTM сети присутствует два вектора трансфера между повторяющимися модулями сети, когда в обычной рекуррентной сети лишь один. Это позволяет передавать 2 блока данных: вывод текущего модуля и информацию, кото-

рая транслируется через каждую ячейку сети с минимальными изменениями. А это значит, что проблема с воспроизведением далеких решений сети от текущей задача, решена.

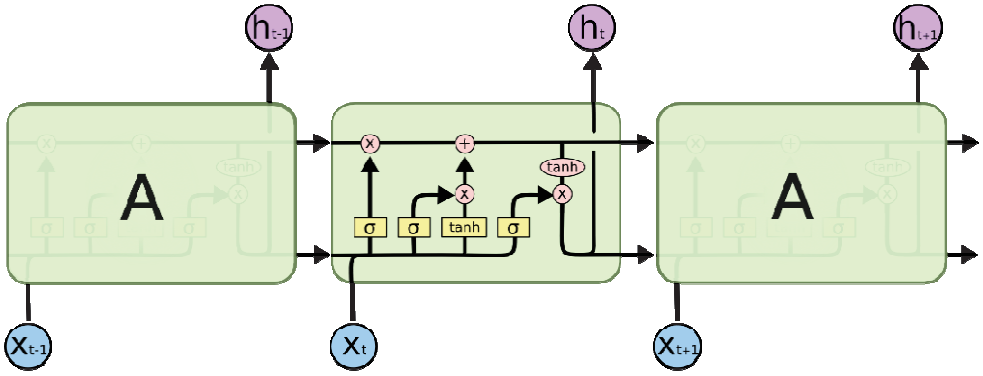

Рисунок 2 – Схема LSTM сети

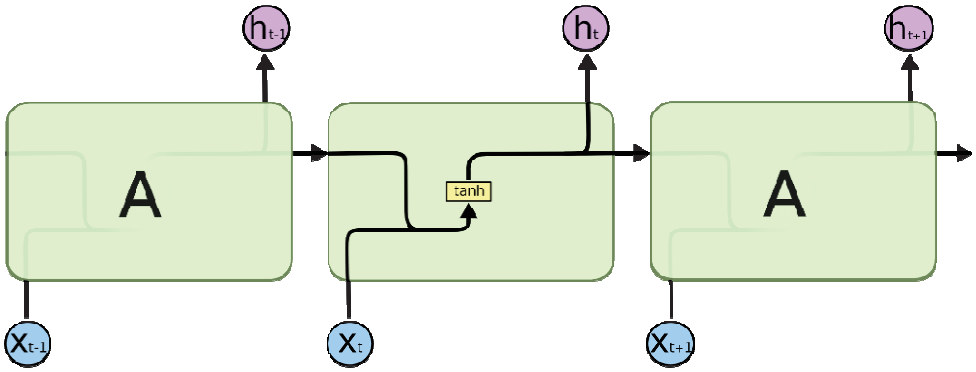

Рисунок 3 – Схема RNN сети

Эффективность LSTM сетей выше, чем эффективность обычных рекуррентных сетей за счет долгого сохранения памяти, что было подтверждено при реализации проекта: точность данного типа сети колебалась от 80 до 95 процентов, в зависимости от размера обучающей выборки.

ЛИТЕРАТУРА

1. Официальная документация Keras: https://keras.io/
2. Официальная документация Tensorflow: https://tensorflow.org/
3. Jason Brownlee – Long Short-Term Memory Networks With Python. eBook. 246 с.