Лабораторная работа 9

Анализ Фишинга моделями глубокого обучения

Дан датасет Фишинга

https://www.kaggle.com/datasets/taruntiwarihp/phishing-site-urls

Разработать и протестировать модели глубокого обучения для классификации фишинга на основе датасета. Провести сравнительный анализ различных алгоритмов.

Этапы выполнения

1. Подготовка данных

1. **Загрузка данных**
   - Скачать датасет с Kaggle и загрузить в среду разработки (Google Colab, Jupyter Notebook, PyCharm).
   - Использовать pandas и numpy для работы с данными.
2. **Анализ данных**
   - Определить целевую переменную (метка фишинговый сайт / безопасный сайт).
   - Проверить баланс классов (value_counts()).
   - Изучить текстовые признаки (например, URL, доменное имя, параметры запроса).
   - Проверить наличие пропущенных значений (df.isnull().sum()).
3. **Предобработка данных**
   - Заполнить или удалить пропущенные значения.
   - Очистить текстовые данные (удаление специальных символов, нормализация).
   - Преобразовать текст в числовой формат (TF-IDF, Word2Vec, FastText, CountVectorizer).
   - Разделить данные на обучающую и тестовую выборки (train_test_split).
   - Преобразовать данные в формат, подходящий для нейросетей (reshape, to_categorical для целевой переменной).

2. Обучение нейронных сетей

Обучить и протестировать три типа нейронных сетей:

2.1 Полносвязная нейронная сеть (Dense Neural Network, DNN)

Архитектура:

- Входной слой (Input Layer).
- Несколько скрытых слоев с Dense и ReLU.
- Dropout (для предотвращения переобучения).
- Выходной слой с sigmoid (если бинарная классификация) или softmax (если многоклассовая).

Библиотеки:

- TensorFlow/Keras
- Dense из tf.keras.layers

Гиперпараметры для настройки:

- Количество слоев и нейронов.
- learning_rate (оптимизатор Adam).
- batch_size, epochs.

2.2 Сверточная нейронная сеть (Convolutional Neural Network, CNN)

Архитектура:

- Входной слой (Input Layer), преобразующий данные в 2D-матрицу.
- Embedding слой для представления символов и слов в векторном виде.
- Conv1D слои для обработки текстовой информации.
- BatchNormalization и ReLU для улучшения сходимости.
- MaxPooling1D для снижения размерности.
- Flatten и Dense для классификации.

Библиотеки:

- Embedding, Conv1D, MaxPooling1D, Flatten, Dense из tf.keras.layers.

Гиперпараметры:

- Количество фильтров и размер ядра в Conv1D.
- Размерность MaxPooling1D.
- Количество Dense-слоев.

2.3 Рекуррентная нейронная сеть (Recurrent Neural Network, RNN)

Архитектура:

- Embedding слой для представления слов в виде плотных векторов.
- LSTM или GRU слой для обработки последовательности символов URL-адреса.
- Dropout и BatchNormalization для регуляризации.
- Dense слой для классификации.

Библиотеки:

- Embedding, LSTM, GRU из tf.keras.layers.

Гиперпараметры:

- Количество LSTM-нейронов.
- Количество слоев LSTM.
- Размер batch_size.

3. Оценка моделей

1. **Метрики качества**

    - accuracy
    - precision
    - recall
    - F1-score
    - ROC-AUC
2. **Кросс-валидация**
    - Использование KFold или StratifiedKFold.
3. **Визуализация обучения**
    - Графики loss и accuracy по эпохам.

4. Анализ и выводы

1. Сравнить результаты всех моделей:
    - Какая архитектура работает лучше?
    - Время обучения каждой модели.
    - Как обработка входных данных влияет на результат?
2. Сделать выводы о применимости нейросетевых моделей к задаче классификации фишинговых сайтов.

5. Требования к отчету

1. Код с комментариями.
2. Графики и таблицы с результатами.
3. Описание результатов и выводы.