ПРИМЕНЕНИЕ АППАРАТА НЕЙРОСЕТЕВОГО МОДЕЛИРОВАНИЯ ДЛЯ АНАЛИЗА И ПРОГНОЗИРОВАНИЯ КРИПТОВАЛЮТ НА ФИНАНСОВЫХ БИРЖАХ

Д. Р. Алипчикова, Г. А. Петрусевич

Научный руководитель – Г. А. Хацкевич, доктор экономических наук, профессор

В статье рассматривается использование нейросетевого моделирования в анализе и прогнозировании криптовалют на финансовых биржах. Авторы предлагают два подхода: использование глубоких нейронных сетей для обучения на исторических данных о ценах на криптовалюты и использование рекуррентных нейронных сетей для анализа временных рядов цен. Также в статье рассматривается роль криптовалютных бирж и объясняется, что они предоставляют пользователям возможность торговать криптовалютами и зарабатывать на изменении их цен

Ключевые слова: нейросетевое моделирование; криптовалюты; финансовые биржи; прогнозирование; глубокие нейронные сети; рекуррентные нейронные сети; исторические данные; цены; криптовалютные биржи; торговля.

ВВЕДЕНИЕ

Аппарат нейросетевого моделирования может быть полезным инструментом для анализа и прогнозирования криптовалют на финансовых биржах.

Одним из способов применения нейросетевого моделирования в анализе и прогнозировании криптовалют является использование глубоких нейронных сетей для обучения на исторических данных о ценах на криптовалюты. Другой способ использования нейросетевого моделирования – это использование рекуррентных нейронных сетей для анализа временных рядов цен на криптовалюты. Эти сети могут использоваться для прогнозирования будущих цен на основе предыдущих значений цен и других факторов, таких как торговый объем, новости и т.д.

Криптовалютная биржа – это онлайн-платформа, которая позволяет пользователям покупать, продавать и обменивать криптовалюты. Они обычно работают круглосуточно и доступны для пользователей из разных стран мира. Их суть и смысл заключается в том, что они предоставляют пользователям возможность торговать криптовалютами и зарабатывать на изменении их цен. Биржи обеспечивают ликвидность для криптовалют и создают условия для формирования цен на основе спроса и предложения.

Нелинейные технологии моделирования криптовалют включают в себя аппарат нейросетевого моделирования, который является мето-

дом анализа данных, основанным на искусственных нейронных сетях. Нейросетевое моделирование позволяет анализировать сложные зависимости между различными переменными и выявлять скрытые паттерны в данных.

РЕЗУЛЬТАТЫ И ИХ ОБСУЖДЕНИЕ

Одним из примеров применения нейросетевого моделирования к криптовалютам является прогнозирование цены биткоина. На основе этой модели можно прогнозировать будущие цены на биткоин, однако цены на криптовалюту могут подвергаться воздействию различных факторов, таких как новости, регулирование, изменения технологий и т.д.

Как было упомянуто ранее, рекуррентные нейронные сети (RNN) могут быть использованы для моделирования последовательностей данных, таких как курс биткоина. В этом случае, данные по курсу биткоина и другим важным факторам могут быть использованы для обучения RNN, чтобы получить прогнозы будущих значений. Кроме того, можно использовать другие модели машинного обучения, такие как градиентный бустинг, случайный лес и т.д. В зависимости от данных и задачи, некоторые модели могут работать лучше, чем другие.

За всеми архитектурами нейронных сетей уследить сложно, поэтому рассмотрим одну из самых популярных – Gated recurrent units (GRU). По сути, это разновидность архитектуры LSTM. Стоит отметить, что GRU уже реализовано в Keras, поэтому остается только импортировать нужные библиотеки.

```
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
from sklearn.preprocessing import MinMaxScaler
from keras.models import Sequential
from keras.layers import Dense, Dropout, GRU
from keras.optimizers import SGD
```

Загружается датасет, далее необходимо перевести дату к Datetime формату, а строковые цены к float, затем строится график зависимости курса биткоина от времени (результат на рисунке 1).

```
df = pd.read_csv('BTC_USD.csv')
df = df[['Дата', 'Цена']]
df['Дата'] = pd.to_datetime(df['Дата'], format = '%d.%m.%Y')
df = df.sort_values('Дата').reset_index(drop=True)
df['Цена'] = df['Цена'].apply(lambda x: x.replace('.', '').split(',')[0])
df['Цена'] = df['Цена'].astype(float)
print(df.shape)
plt.figure(figsize = (20, 7))
```

```
plt.plot(df['Дата'], df['Цена'], color = 'g')
plt.xlabel('Дата')
plt.ylabel('Цена, USD')
```

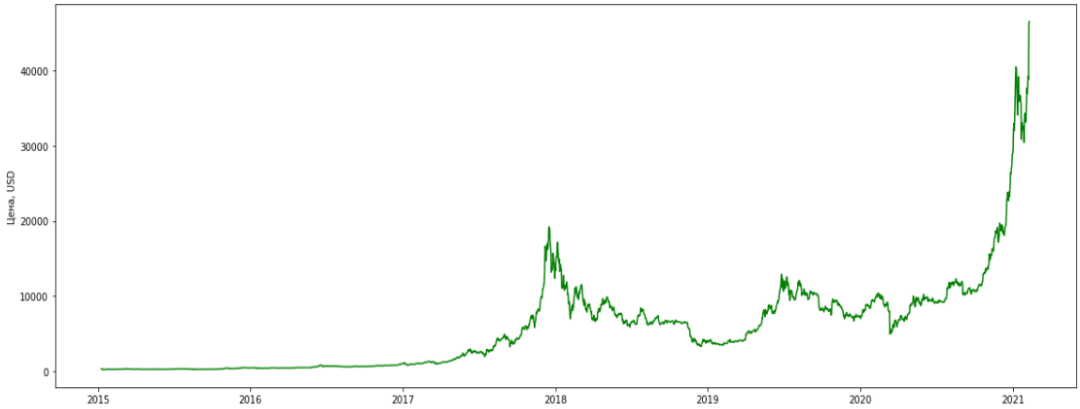

Рис. 1. Результат выполнения кода

Далее данные разделяются на тренировочную и тестовую выборки, данные нормализуются в масштабе от 0 до 1.

```
split_num = 2150
train = df.iloc[:split_num, 1:2]
test = df.iloc[split_num:, 1:2]
scaler = MinMaxScaler(feature_range = (0, 1))
train_scaled = scaler.fit_transform(train)
```

То есть предсказание будет выдавать курс на 61 день.

```
X_train = []
y_train = []
window = 60
for i in range(window, split_num):
    X_train_ = np.reshape(train_scaled[i - window:i, 0], (window, 1))
    X_train.append(X_train_)
    y_train.append(train_scaled[i, 0])
X_train = np.stack(X_train)
y_train = np.stack(y_train)
```

Стоит отметить, что у первого предсказанного дня уже будет какая-то ошибка и даже если она мала, она имеет свойство накапливаться. Из этого следует вывод о том, что такие модели хороши для предсказаний на короткий период времени.

Тестовая модель обучается на 1000 эпох, минимизируется MSE, от переобучения добавляется дропаут, а также устанавливается размер батча равным 16.

```
model = Sequential()
model.add(GRU(units = 50, return_sequences = True, input_shape=(X_train.shape[1],1)))
model.add(Dropout(0.2))
model.add(GRU(units = 50, return_sequences = True, input_shape=(X_train.shape[1],1)))
model.add(Dropout(0.2))
model.add(GRU(units = 50, return_sequences = True, input_shape=(X_train.shape[1],1)))
model.add(Dropout(0.2))
model.add(GRU(units = 50))
model.add(Dropout(0.2))
model.add(Dense(units = 1))
```

```
model.compile(optimizer = 'sgd', loss = 'mean_squared_error')
model.fit(X_train, y_train, epochs = 1000, batch_size = 16, verbose = 0)
```

Далее тестовые данные преобразовываются.

```
df_volume = np.vstack((train, test))
inputs = df_volume[df_volume.shape[0] - test.shape[0] - window:]
inputs = inputs.reshape(-1,1)
inputs = scaler.transform(inputs)
num_2 = df_volume.shape[0] - split_num + window
X_test = []
for i in range(window, num_2):
    X_test_ = np.reshape(inputs[i-window:i, 0], (window, 1))
    X_test.append(X_test_)
X_test = np.stack(X_test)
```

Предсказания и график на последние пару месяцев, результат на рисунке 2.

```
predict = model.predict(X_test)
predict = scaler.inverse_transform(predict)
plt.figure(figsize = (20, 7))
plt.plot(df['Дата'][2000:], df_volume[2000:], color = 'g', label = 'Real')
plt.plot(df['Дата'][-predict.shape[0]:].values, predict, color = 'r', label = 'Predicted')
plt.xlabel('Дата')
plt.ylabel('Цена, USD')
plt.legend()
```

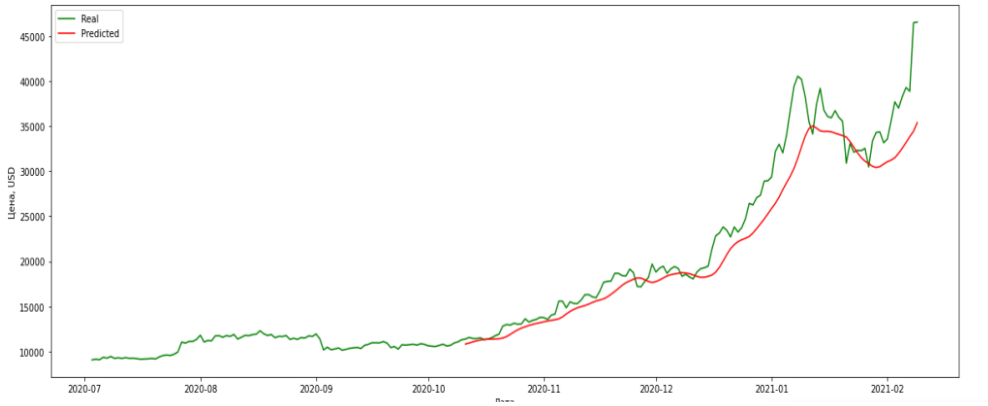

Рис. 2. Результат выполнения кода

Бейзлайн модель довольно точно описывает тренды, но получается достаточно гладкой, без явных скачков.

```
pred_ = predict[-1].copy()
prediction_full = []
window = 60
df_copy = df.iloc[:, 1:2][1:].values
for j in range(20):
    df_ = np.vstack((df_copy, pred_))
    train_ = df_[:split_num]
    test_ = df_[split_num:]
    df_volume_ = np.vstack((train_, test_))
    inputs_ = df_volume_[df_volume_.shape[0] - test_.shape[0] - window:]
    inputs_ = inputs_.reshape(-1,1)
    inputs_ = scaler.transform(inputs_)
    X_test_2 = []
```

```
        for k in range(window, num_2):
            X_test_3 = np.reshape(inputs_[k - window:k, 0], (window, 1))
            X_test_2.append(X_test_3)
        X_test_ = np.stack(X_test_2)
        predict_ = model.predict(X_test_)
        pred_ = scaler.inverse_transform(predict_)
        prediction_full.append(pred_[-1][0])
        df_copy = df_[j:]
    prediction_full_new = np.vstack((predict, np.array(prediction_full).reshape(-1,1)))
    df_date = df[['Дата']]
    for h in range(20):
        kk = pd.to_datetime(df_date['Дата'].iloc[-1]) + pd.DateOffset(days=1)
        kk = pd.DataFrame([kk.strftime("%Y-%m-%d")], columns = ['Дата'])
        df_date = df_date.append(kk)
    df_date = df_date.reset_index(drop = True)
    df_date['Дата'] = pd.to_datetime(df_date['Дата'])
```

Нарисуем финальный график, результат на рисунке 3.

```
    plt.figure(figsize = (20,7))
    plt.plot(df['Дата'][2000:], df_volume[2000:], color = 'red', label = 'Real')
    plt.plot(df_date['Дата'][-prediction_full_new.shape[0]:].values, prediction_full_new, color = 'blue', label = 'Predicted')
    plt.xlabel('Дата')
    plt.ylabel('Цена, USD')
    plt.legend()
```

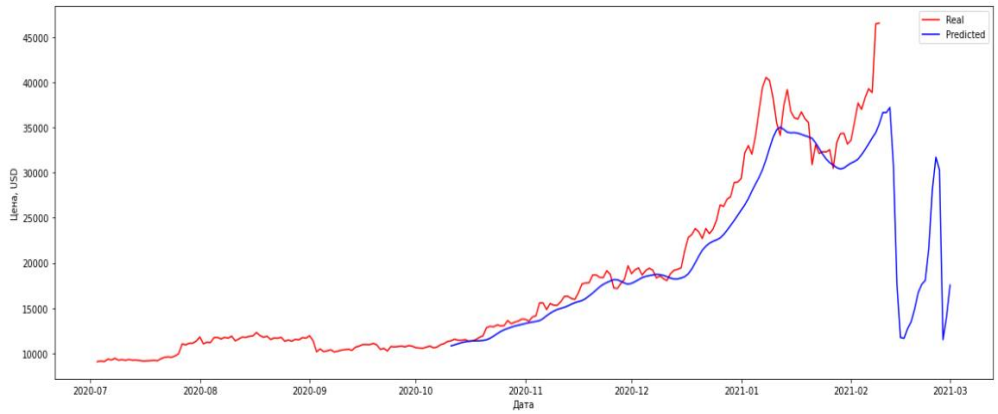

Рис. 3. Результат выполнения кода

Модель предсказывает падение курса биткоина до уровня 2019 года. В данной статье получается бейзлайн для задачи прогнозирования временных рядов, для ее улучшения нужно экспериментировать с архитектурой сети, с различными гиперпараметрами, как самой сети, так и некими переменными для времени (ширина окна, количество дней для предсказания).

Библиографические ссылки

1. *Лебедев А. С., Смирнов А. И.* Прогнозирование криптовалютных цен с помощью нейросетей // Информационные технологии и вычислительные системы. 2019. Т. 17, № 1. С. 122–128.

2. *Максимов Д. В., Комлев А. Н., Герасимов Д. И.* Применение глубоких нейронных сетей для прогнозирования цен на криптовалюты // Информационные технологии и вычислительные системы. 2018. Т. 16, № 4. С. 108–115.
3. *Луценко В. А., Павлов А. А., Иванов А. В.* Прогнозирование цен на криптовалюты с использованием рекуррентных нейронных сетей // Информационные технологии и вычислительные системы. 2020. Т. 18, № 3. С. 95–100.
4. *Медведев Е. В., Поляков О. С., Карпов И. В.* Анализ и прогнозирование криптовалютных рынков на основе нейронных сетей // Информационные технологии и вычислительные системы. 2021. Т. 19, № 2. С. 120–127.