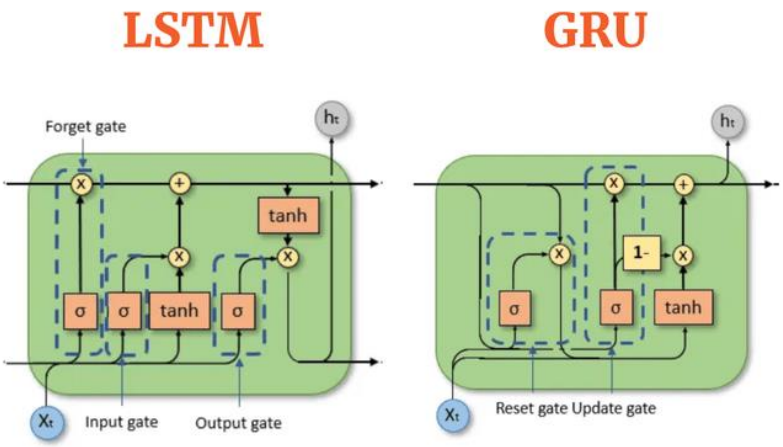
Практическая работа №1

Рекуррентные нейронные сети. Модели LSTM, GRU

Цель работы: изучение работы искусственных рекуррентных нейронный сетей, в частности LSTM и GRU моделей.

Инструментарий: Язык Python (с использованием IDE (PyCharm, Jupyter, Spyder и т.п.), а также библиотек, содержащих которые нейросетевые модели (Tensorflow, Theano, Keras и т.п.). *//опционально, можно согласовать иной инструмент*



Задача №1.

Обучить и использовать LSTM и GRU сети для генерации текста:

- В стиле Чехова А.П.

- Маяковского В.В.

- Достоевского Ф.М.

Тексты для обучения прилагаются к данному документу. Допустимо сформировать иной датасет, содержащий соответствующие тексты. Самостоятельно предобработать и нормализовать (при необходимости) датасет.

Заполнить таблицу для каждой из нейросетевых моделей, в которой включить наборы параметров настройки, характерные для моделей, для каждого из примеров обучения и использования сети (эксперимента).

Таблица 1 –LSTM сеть (пример)

№ п/п	LSTM слой	Слой эмбендинга слов	Выходной слой	Функция активации	Функция потерь	Функция оптимизации	Метрика
1	64, dropout = 0,2, recurrent_dropout= 0.2	max_features, 128	1	sigmoid	binary_crossentropy	adam	accuracy
	…	…	…	…	…	…	…
	…	…	…	…	…	…	…
	…	…	…	…	…	…	…

Таблица 2 – работа LSTM сети (пример)

№ п/п	Количество эпох	Размер бэтча	Метрика
1	5	12	Accuracy = 0,071
…	…	…	…

//параметры могут несколько отличаться, например некоторые параметры можно добавлять, метрики можно использовать различные, в том числе MSE, SE, RMSE

//Т.о. исследования должны проводиться для шести обученных моделей. LSTM и GRU на выборке для стиля каждого из трех авторов. 12 таблиц.

Аналогичные действия произвести в отношении трех текстов, описывающих заболевание: Корь, Чесотка, Шизофрения.

Примеры LSTM и GRU сетей на Python:

Основная LSTM модель:

```
from keras.models import Sequential
from keras.layers import LSTM, Dense

model = Sequential()
model.add(LSTM(50, input_shape=(100, 1)))  # 100 time steps, 1 feature
model.add(Dense(1))
model.compile(optimizer='adam', loss='mse')
```

LSTM с множественными слоями:

```
model = Sequential()
model.add(LSTM(50, return_sequences=True, input_shape=(100, 1)))
model.add(LSTM(50))
model.add(Dense(1))
model.compile(optimizer='adam', loss='mse')
```

LSTM для классификации:

```
model = Sequential()
model.add(LSTM(100, input_shape=(100, 1)))
model.add(Dense(10, activation='softmax'))
model.compile(optimizer='adam', loss='categorical_crossentropy', metrics=['accuracy'])
```

Стековая LSTM:

```
model = Sequential()
model.add(LSTM(100, return_sequences=True, input_shape=(100, 1)))
model.add(LSTM(100))
model.add(Dense(1, activation='sigmoid'))
model.compile(optimizer='adam', loss='binary_crossentropy', metrics=['accuracy'])
```

LSTM с регуляризацией Dropout:

```
from keras.layers import Dropout

model = Sequential()
model.add(LSTM(100, input_shape=(100, 1)))
model.add(Dropout(0.5))
model.add(Dense(1, activation='sigmoid'))
model.compile(optimizer='adam', loss='binary_crossentropy', metrics=['accuracy'])
```

Основная GRU модель:

```
from keras.layers import GRU

model = Sequential()
model.add(GRU(50, input_shape=(100, 1)))
model.add(Dense(1))
model.compile(optimizer='adam', loss='mse')
```

GRU с множественными слоями:

```
model = Sequential()
model.add(GRU(50, return_sequences=True, input_shape=(100, 1)))
model.add(GRU(50))
model.add(Dense(1))
model.compile(optimizer='adam', loss='mse')
```

GRU для классификации:

```
model = Sequential()
model.add(GRU(100, input_shape=(100, 1)))
model.add(Dense(10, activation='softmax'))
model.compile(optimizer='adam', loss='categorical_crossentropy', metrics=['accuracy'])
```

Стековая GRU:

```
model = Sequential()
model.add(GRU(100, return_sequences=True, input_shape=(100, 1)))
model.add(GRU(100))
model.add(Dense(1, activation='sigmoid'))
model.compile(optimizer='adam', loss='binary_crossentropy', metrics=['accuracy'])
```

GRU с регуляризацией Dropout:

```
model = Sequential()
model.add(GRU(100, input_shape=(100, 1)))
model.add(Dropout(0.5))
model.add(Dense(1, activation='sigmoid'))
model.compile(optimizer='adam', loss='binary_crossentropy', metrics=['accuracy'])
```

Задача №2

С использованием LSTM и GRU моделей произвести прогноз состояния здоровья пациента - наличие у него сахарного диабета.

Обучающая выборка имеет следующие поля:

Диабет_012 0 = отсутствие диабета 1 = преддиабет 2 = сахарный диабет,

Высокий уровень (давление) 0 = нет высокого ДАВЛЕНИЯ 1 = высокое ДАВЛЕНИЕ

Высокий холестерин 0 = нет высокого уровня холестерина 1 = высокий уровень холестерина

Проверка холестерина 0 = не проверял уровень холестерина в течение 5 лет 1 = проверял уровень холестерина в течение 5 лет

ИМТ Индекс массы тела

Курильщик Выкурили ли вы за всю свою жизнь хотя бы 100 сигарет? [Примечание: 5 пачек = 100 сигарет] 0 = нет 1 = да

Инсульт (Вам когда-нибудь говорили), что у вас был инсульт. 0 = нет 1 = да

Сердечное заболевание Ишемическая болезнь сердца (ИБС) или инфаркт миокарда (ИМ) 0 = нет 1 = да

Физическая активность физическая активность за последние 30 дней — не включая работу 0 = нет 1 = да

Фрукты употребление фруктов 1 или более раз в день 0 = нет 1 = да

Прогнозируемым параметром является **Диабет_012.**

Привести результаты обучения, тестирования и прогноза при различных наборах параметров, заполнить таблицы аналогичные таблицам в задаче №1.

Построить графики (обязательно отображение изменения прогнозируемого параметра, см. источник [5]).

Задача №3 (дополнительная)

С использованием LSTM и GRU моделей произвести прогноз матча по крикету.

Датасет, состоящий из восьми полей, приложен к данному документу. Самостоятельно предобработать и нормализовать (при необходимости) датасет. Сеть в качестве ответа указывает только победителя матча.

Входными данными для прогноза будут являться названия команд (Team 1, Team 2), место локации (Ground), тип матча (Format).

Примечание: Test – формат проведения матчей по крикету (самый престижный, традиционный), ODI – однодневный международный матч (один из трех форматов), T20I - Twenty20 International разновидность турниров сокращенного формата.

Пример ввода: Team 1 = Russia, Team 2 = USA, Ground = Kirov, Format = ODI.

Ответ сети: Russia

Отчет должен содержать:

- титульный лист

- краткие теоретические сведения, в том числе архитектуру ИНС

- таблицы, со сравнением нейросетевых моделей

- экранные формы работы моделей

- исходный код

- датасет (в частности демонстрация разделения на обучающий и тестовый, задачи 2 и 3)

- выводы.

Список используемых источников:

1. LSTM и GRU URL: https://habr.com/ru/companies/mvideo/articles/780774/
2. PyTorch LSTM: Text Generation Tutorial URL: https://closeheat.com/blog/pytorch-lstm-text-generation-tutorial
3. Прогнозирование временных рядов с помощью рекуррентных нейронных сетей URL: https://habr.com/ru/articles/495884/
4. Прогнозирование временных рядов с использованием нейронных сетей LSTM: Нормализация цены и токенизация времени URL: https://www.mql5.com/ru/articles/15063
5. Прогнозирование курса акций с использованием LSTM и его реализация URL: https://www.analyticsvidhya.com/blog/2021/12/stock-price-prediction-using-lstm/
6. Генерация текста с использование м LSTM URL: https://coderzcolumn.com/tutorials/artificial-intelligence/pytorch-text-generation-using-lstm-networks