ОЛИМПИАДА ШКОЛЬНИКОВ «ШАГ В БУДУЩЕЕ»

НАУЧНО-ОБРАЗОВАТЕЛЬНОЕ СОРЕВНОВАНИЕ «ШАГ В БУДУЩЕЕ, МОСКВА»

2922

регистрационный номер

Информатика и системы управления (ИУ)

название факультета

Программное обеспечение ЭВМ и информационные технологии (ИУ-7)

название кафедры

Умный аранжировщик

название работы

Автор: **Шумнов Пётр Алексеевич**

фамилия, имя, отчество

ГБОУ Школа №1533 «ЛИТ»

наименование учебного заведения, класс

Научный руководитель: **Завриев Николай Константинович**

фамилия, имя, отчество

ГБОУ Школа №1533 «ЛИТ»

место работы

Преподаватель по информатике

звание, должность

_____

подпись научного руководителя

Москва - 2019

Умный аранжировщик

Аннотация

Цель проекта – разработать программу, которая бы позволяла автоматически генерировать аккомпанемент к гитарной партии. В частности, писала бы партии для барабанов и бас-гитары, требуя при этом от пользователя лишь загрузить свою гитарную партию и выбрать жанр и стиль. Программа может быть использована для создания «минусовок» для игры или же для того, чтобы музыкант мог оценить, как придуманная им партия будет звучать в различных аранжировках.

Так как задача носит творческий характер, было решено использовать методы машинного обучения. Было опробовано более 60 моделей, различающихся методом преобразования входных и выходных данных, типом нейронной сети, количеством слоев и нейронов. На данный момент наилучший результат достигнут при помощи нейронной сети с одним скрытым LSTM-слоем с 50 нейронами. В этой модели партии инструментов представлены как последовательности нот, и каждый такт обрабатывается отдельно.

Конечная точность (accuracy) нейронной сети на валидационной выборке – 80%. Однако этот показатель не позволяет оценить качество работы программы. Дело в том, что данный метод вычисления точности предполагает для каждого входного вектора – партии гитары – единственно верный правильный выходной вектор – партии барабанов и бас-гитары. Соответственно, оценивается приближенность сочиненной нейросетью партии к некому правильному ответу. Однако задача сочинения музыки творческая, «правильных» вариантов аранжировки огромное множество. Так что оценить результат может только человек на слух.

На данный момент программа способна генерировать партии барабанов и бас-гитары в стилях «рок» и «рок`н`ролл».

Оглавление

1 Введение

В настоящее время из-за многих факторов (например, повышение доступности музыкальных инструментов) очень распространена музыкальная деятельность. Научиться играть на музыкальном инструменте при наличии интернета совсем нетрудно и не затратно. При этом одним из самых распространенных инструментов в мире является гитара, а программное обеспечение для гитаристов уже давно стало многомиллионной индустрией и включает в себя огромное количество разного рода приложений.

Однако существующие программы автоматического аккомпанемента для гитары имеют значительные недостатки. В данном проекте применяется принципиально новый в этой области подход – использование машинного обучения для создания аккомпанемента с нуля. Среди аналогов присутствует программа, в которой нейронная сеть подбирает для каждого такта композиции наиболее подходящий шаблон. Однако этот метод не позволяет создавать ничего кроме барабанов, в то время как модель, применяемую в данном проекте, можно обучить играть на любом инструменте в любом стиле.

2 Постановка задачи

Цель проекта – разработать программу, автоматически создающую партии для барабанов и бас-гитары на основе партии гитары. Предполагается, что данная программа будет использоваться вместе с табулатурным редактором Guitar Pro, так как она работает с форматами этого приложения.

На вход программа получает файл формата Guitar Pro любой из трех версий: .gp3, .gp4, .gp5 – содержащий некоторое число дорожек с табулатурами/нотами. Далее пользователь имеет возможность добавить произвольное число новых дорожек и сохранить файл в одном из перечисленных выше форматов.

При добавлении новой дорожки пользователь должен указать следующие ее параметры:

- «Родительскую» дорожку – ту, на основе которой будет генерироваться новая
- Тип: барабаны или бас-гитара
- Желаемый музыкальный жанр/стиль
- Имя

3 Актуальность и целевая аудитория

Табулатура – аналог нотной записи музыки, широко распространенный среди гитаристов. Guitar Pro является одним из самых популярных табулатурных редакторов, предоставляющих пользователю огромный функционал для чтения, воспроизведения и редактирования как табулатур, так и нот.

Целевая аудитория проекта – гитаристы, в частности пользователи Guitar Pro и его аналогов (большинство подобных программ поддерживают формат файлов Guitar Pro). Данный проект может быть использован несколькими способами:

- При помощи программы гитарист может создать «минусовку» для игры.
- Если музыкант придумал гитарную партию, он может посмотреть, как она будет звучать в различных аранжировках.
- В дальнейшем при улучшении качества возможно третье применение: создание аранжировки для конечного продукта. При таком использовании могут помочь VST-плагины, способные «оживить» стандартный звук midi и сделать его приемлемым для конечной записи песни.

4 Обзор аналогов

Данный проект имеет множество аналогов, однако значительно отличается от них как областью применения и целевой аудиторией, так и реализацией.

Все аналоги можно разделить на три группы:

1. **Ручные**, например:
    - Guitar Pro.
    - TuxGuitar.

    В основном это редакторы табулатур или нот. Все партии там создаются вручную и никакой автоматической генерации они не включают. Однако, они могут являться аналогами, т.к. имеют отчасти похожие с данным проектом цели - создание партий инструментов. Тем не менее, это наиболее отдаленные аналоги проекта.
2. **Полуавтоматизированные**, например:
    - LA Scoring Strings.
    - Band-in-a-Box.

    В этих программах процесс создания аккомпанемента автоматизирован лишь частично: пользователю требуется ввести в программу последовательность аккордов, которые уже будут обыгрываться автоматически.

    Такой подход имеет некоторые минусы:
        - Пользователю требуется умение гармонизировать мелодию.
        - Пользователю придется тратить на этот процесс время.
        - Результат всегда будет представлять собой последовательность аккордов.

    Также, не было найдено ни одной подобной программы, позволяющей генерировать ударные.
3. **Автоматизированные**, например:
    - Microsoft SongSmith.

- «Automatic Accompaniment Generation with Seq2Seq».
- GarageBand

Этот вид аналогов ближе всего к данному проекту. Было найдено всего 3 подобные программы. Они позволяют полностью автоматически создавать аккомпанемент, однако тоже имеют ряд отличий от данного проекта. Стоит подробно рассмотреть каждый из них.

Microsoft Songsmith - это программа для создания аккомпанемента к вокалу, и работает она только с ним. Отсутствует возможность генерации барабанов, так как вокал изначально записывается под один из шаблонов ударных. Так же, результат опять же представляет собой только обыгровку аккордов.

«Automatic Accompaniment Generation with Seq2Seq» – это проект Калифорнийского университета в Беркли, основанный на работе Стэнфордского университета. Целью этого проекта было проверить, могут ли нейросети генерировать аккомпанемент к мелодии. Можно сказать, что ответ был получен положительный, однако результат очень примитивен, в особенности ударные.

И, наконец, GarageBand – программа от Apple, которая позволяет автоматически генерировать барабаны. О недостатке данного ПО говорилось во введении – программа лишь подбирает наиболее подходящие «сэмплы» (шаблоны барабанных партий). Таким образом можно сгенерировать только ударные.

После анализа аналогов был выявлен ряд преимуществ данного проекта:

- Полная автоматизация, требующая от пользователя только выбора музыкального жанра.
- Возможность создания партии ударных.
- Гибкость применяемого метода – возможность обучить модель сочинять партии любых инструментов в любом жанре.

5 Математические модели и алгоритмы

В данном проекте было решено использовать методы машинного обучения, так как задача носит творческий характер. Сначала было опробовано использовать многослойный перцептрон, однако не удалось добиться приемлемого результата. Поэтому следующим методом было глубокое обучение, в частности, рекуррентные нейронные сети (РНС).

Для применения РНС входные данные и выходные данные – партии инструментов – представляются в виде последовательностей нот. Данная модель называется (sequence-to-sequence, последовательность к последовательности). Подробное описание принципа работы РНС представлено в приложении А.

Несмотря на все преимущества, которые дает возможность «запоминать» информацию, у РНС есть большой недостаток: проблема долговременных зависимостей. Нейроны хорошо «помнят» недавно полученную информацию, но не имеют возможности надолго сохранить в памяти что-то, что обработали много циклов назад, какой бы важной та информация ни была. Для решения этой проблемы используют частный вид РНС – LSTM (Long short-term memory, долгая краткосрочная память). Это особая разновидность архитектуры рекуррентных нейронных сетей, способная к обучению долговременным зависимостям.

В данном проекте LSTM-сети показали себя лучше, чем классические РНС. В приложении Б представлено описание принципа работы LSTM-сети.

6 Обученные модели

На данный момент для генерации барабанов было опробовано 66 моделей, различающихся входными/выходными данными, типом нейросети и количеством слоев и нейронов. Самым главным параметром является способ преобразования входных и выходных данных – партий инструментов – в векторы, пригодные для нейросети. По данному параметру все 66 моделей можно разделить на 5 общих категорий:

1. Самый простой и очевидный вариант, который на данный момент и показал себя лучше всех. Использованы LSTM-сети, за единицу последовательности взята одна нота и каждый такт песни анализируется отдельно и независимо. Также, для этого варианта пробовалось использовать многослойный перцептрон.
2. Аналогичный вариант, но анализируется сразу вся песня – последовательность состоит из нот не одного такта, а всего произведения.
3. Снова LSTM-сети, но за единицу последовательности взята не нота, а целый такт. Песня анализируется целиком.
4. Задача делится на 2 части: сначала по партии гитары первая нейросеть выдает только ритм, а затем вторая по этому ритму пишет партию ударных. Это была попытка упростить задачу, ведь нейросети не нужно будет учиться и выделять ритм, и писать партию конкретно барабанов. Было опробовано брать за единицу последовательности как ноту, так и такт, как в первом и третьем вариантах. Для каждой из двух нейросетей были опробованы 2 типа: LSTM и многослойный перцептрон.
5. Партия барабанов разделялась на партии каждого барабана отдельно, и обучалось 6 нейросетей, каждая училась играть на определенном барабане. Далее по определенному алгоритму эти партии объединялись.

Также, были опробованы разные способы описания ноты. В конечном итоге гитарная (входная) нота описывается 3 параметрами:

- Нота/пауза – логическая переменная

- Количество сыгранных струн. Введение этого параметра позволило определить, была сыграна отдельная нота или аккорд, что сильно улучшило результат.
- Длительность ноты.

Добавлялись такие параметры, как:

- Высота ноты.
- Разность высот текущей и предыдущей нот.

Нота барабанов описывает 7 параметрами: первый – длительность, каждый из следующих 6 отвечает за какой-либо барабан и равен 1, если удар в барабан был, и 0, если нет.

Для каждого их этих вариантов варьировалось также количество слоев и нейронов в сети. Были попытки заменять LSTM на GRU и классическую RNN, но к улучшению результатов это не привело.

В большинстве случаев нейронная сеть почти не обучалась (находила лишь самые простые закономерности, как, например, выделение первой доли), либо работала хуже первого варианта.

Для бас-гитары входные данные преобразуются следующим образом: все ноты «переносятся» в одну октаву, далее нейросети даются те же параметры нот, что и нейросети для барабанов, с добавлением высоты (число от 0 до 11 – в пределе одной октавы). Выходная же нота имеет 3 параметра: нота/пауза, длительность и высота ноты.

7 Программная реализация

Для разработки программы использовался язык python, так как он считается основным языком для работы с нейросетями. На рис. 1 изображена схема компонентов программы.

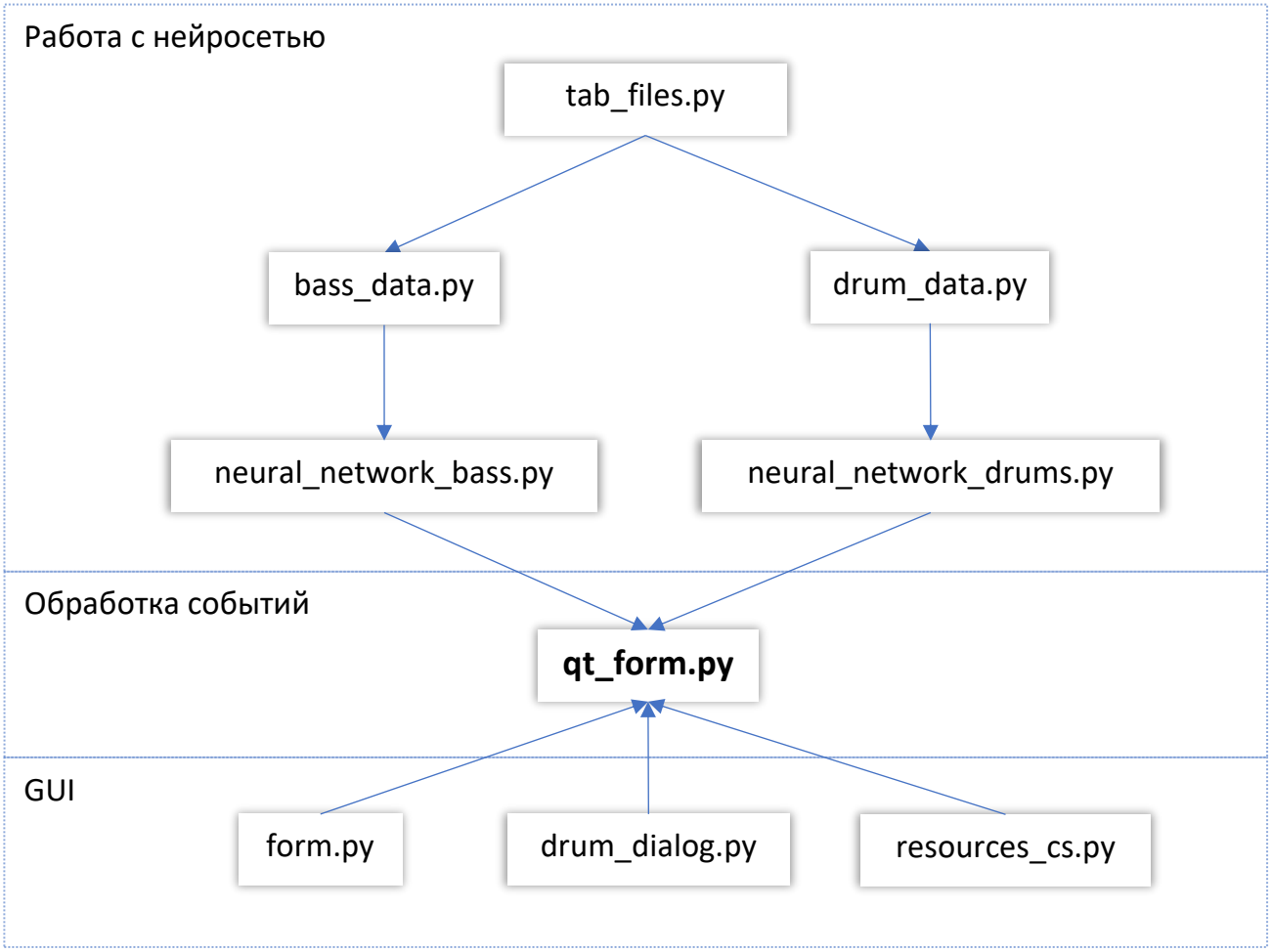

Рис. 1: компоненты программы

Программу можно условно разделить на 3 блока:

1 Графический интерфейс пользователя.

При создании графического интерфейса использовались среда Qt Designer для разработки дизайна и библиотека pyqt5 для импорта и работы с ним в python-коде.

В данный блок можно включить 3 файла, которые были сгенерированы автоматически при экспорте из Qt Designer:

- Form.py – файл с дизайном основной формы.
- Drum_dialog.py – файл с дизайном диалогового окна создания новой дорожки.

- Resources_cs.py – файл с ресурсами.

2 Обработка событий.

В этом блоке описаны все взаимодействия пользователя с программой. Он состоит из одного файла – qt_form.py – который запускается первым при открытии программы.

3 Работа с нейросетью.

Самая сложная часть программы, на создание которой ушла подавляющая часть времени работы над проектом. Блок включает 5 файлов:

- Tab_files.py – файл, в котором описаны классы для работы с табулатурами (см рис. 15).
- Bass_data.py и Drum_data.py – файлы, в которых происходит подготовка входных данных (как для обучения, так и для валидации и тестирования) и преобразование выхода нейросети в табулатуру.
- Neural_network_bass.py и Neural_network_drums – файлы с кодом нейросетей (для бас-гитары и барабанов соответственно).

Список используемых библиотек:

- Keras + TensorFlow для реализации нейросети.
- PyGuitarPro для чтения файлов табулатур.
- Numpy для работы с массивами.
- MsgPack для сериализации данных.

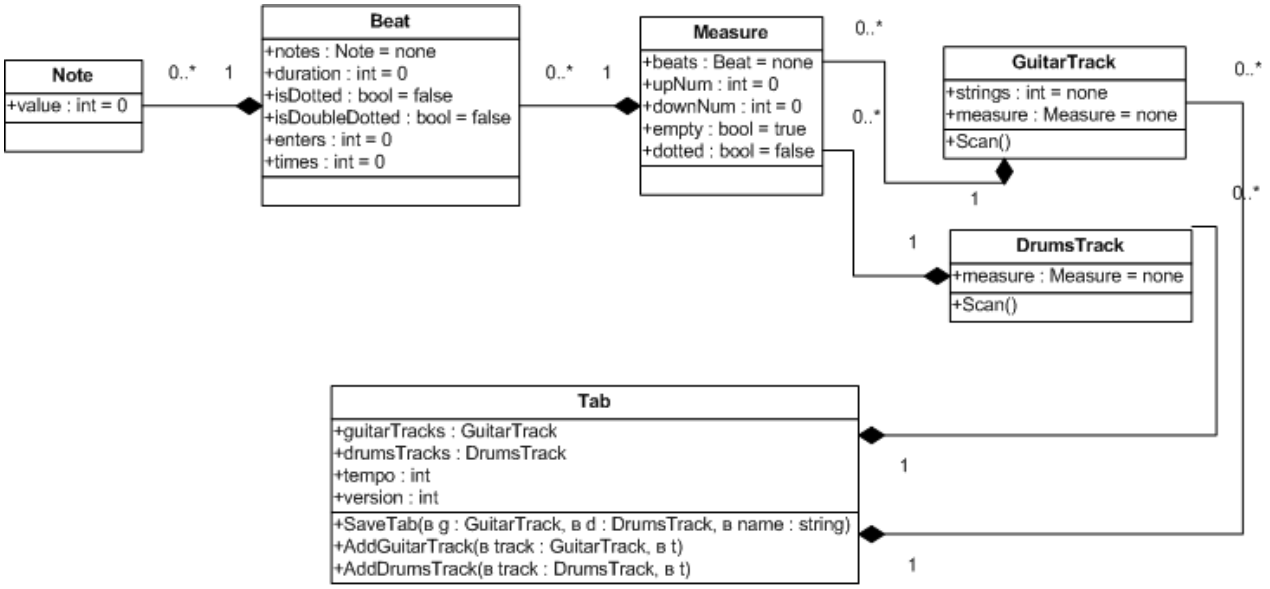

Рис. 15: диаграмма классов файла tab_files.py

8 Результаты

Поставленная задача была реализована частично: реализована генерация барабанов и бас-гитары в двух жанрах: «рок» и «рок`н`ролл».

Однако благодаря гибкости разработанной системы, для добавления новых жанров требуется лишь обучить уже существующую модель на новых данных и поместить файл с весами в папку с проектом.

На рис. 2 изображены 2 графика: синий – accuracy (точность), оранжевый – loss (ошибка). По горизонтальной оси отложены эпохи от начала обучения. Видно, что ошибка падает, что является основным признаком успешного обучения нейросети.

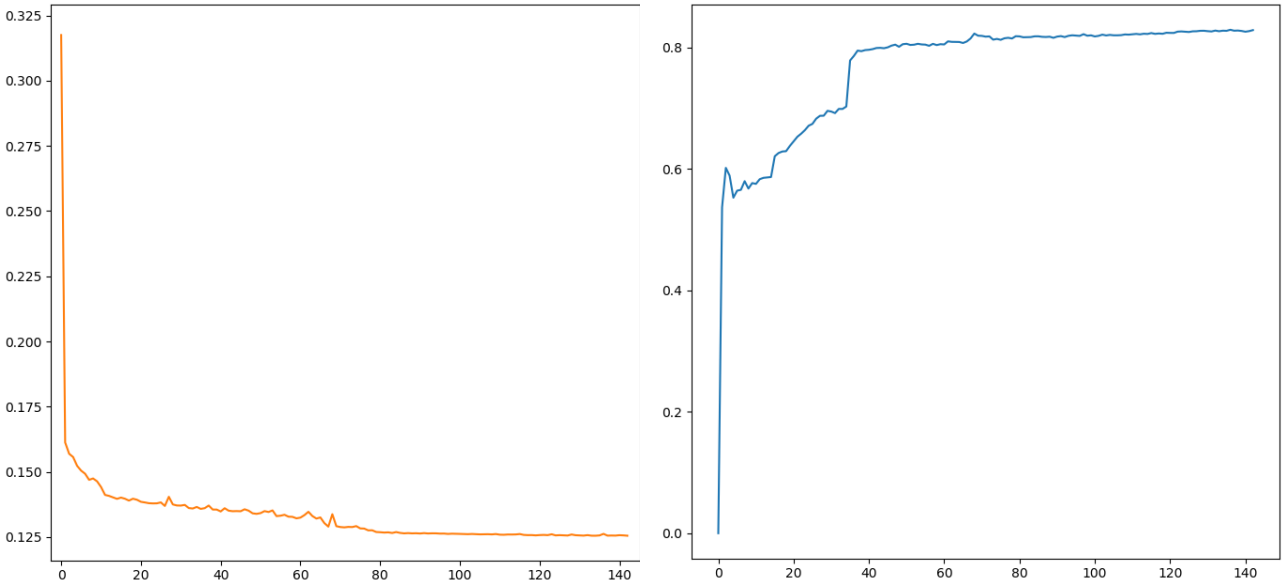

Рис. 2: график loss (ошибки) – оранжевым и accuracy (точности) - синим

Итоговая точность сети ~80% для бас-гитары и ~65% для барабанов. Однако эта оценка весьма условна, так как задача носит творческий характер. Для одной партии гитары может быть сочинено огромное количество разных удовлетворительных партий барабанов, в то время как подобные методы предполагают только один правильный ответ. Таким образом, оценить работу программы может только человек на слух.

9 Заключение

По итогам проделанной работы можно сказать, что нейронные сети определенно способны справиться с поставленной задачей. Предложенный метод превосходит по возможностям используемые в аналогах и имеет большой потенциал. В дальнейшем результат может быть значительно улучшен благодаря более оптимальному подбору параметров нейросети, корректировке формата входных и выходных данных и увеличению времени обучения.

10 Список литературы

1 Тадеусович, Р. Элементарное введение в технологию нейронных сетей с примерами программ / Р. . Тадеусович. и др. – : Горячая Линия - Телеком, 2011. – 408 с.

2 Melody-to-Chord using paired model and multi-task learning language modeling [Электронный ресурс]. – Режим доступа : https://web.stanford.edu/class/cs224n/reports/2742800.pdf, свободный. – Загл. с экрана.

3 Automatic Accompaniment Generation with Seq2Seq [Электронный ресурс]. – Режим доступа : http://qihqi.github.io/machine/learning/music-generation-using-rnn/, свободный. – Загл. с экрана.

4 Нейронные сети для начинающих. Часть 1 [Электронный ресурс]. – Режим доступа : https://habr.com/post/312450/, свободный. – Загл. с экрана.

5 Нейронные сети для начинающих. Часть 2 [Электронный ресурс]. – Режим доступа : https://habr.com/post/313216/, свободный. – Загл. с экрана.

6 LSTM – сети долгой краткосрочной памяти [Электронный ресурс]. – Режим доступа : https://habr.com/company/wunderfund/blog/331310/, свободный. – Загл. с экрана.

7 Библиотеки для глубокого обучения: Keras [Электронный ресурс]. – Режим доступа : https://habr.com/company/ods/blog/325432/, свободный. – Загл. с экрана.

8 Understanding LSTM Networks [Электронный ресурс]. – Режим доступа : http://colah.github.io/posts/2015-08-Understanding-LSTMs/, свободный. – Загл. с экрана.

9 Keras documentation [Электронный ресурс]. – Режим доступа : https://keras.io, свободный. – Загл. с экрана.

10 PyGuitarPro documentation [Электронный ресурс]. – Режим доступа : https://pyguitarpro.readthedocs.io/en/stable/index.html, свободный. – Загл. с экрана.

11 Приложение А

Рекуррентные нейронные сети

На рисунке 3 изображена схема РНС с одним нейроном, получающей на вход последовательность x_t и дающая на выход последовательность h_t. Рекуррентные сети, в отличие от обычных, имеют обратную связь: синапс,

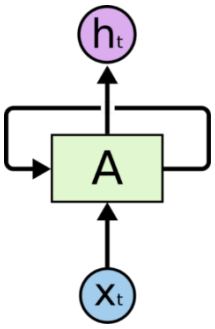

Рис. 3: рекуррентная нейронная сеть с 1 нейроном

который передает информацию от одного шага сети к другому.
Если «развернуть» обратную связь, получим рисунок 4.

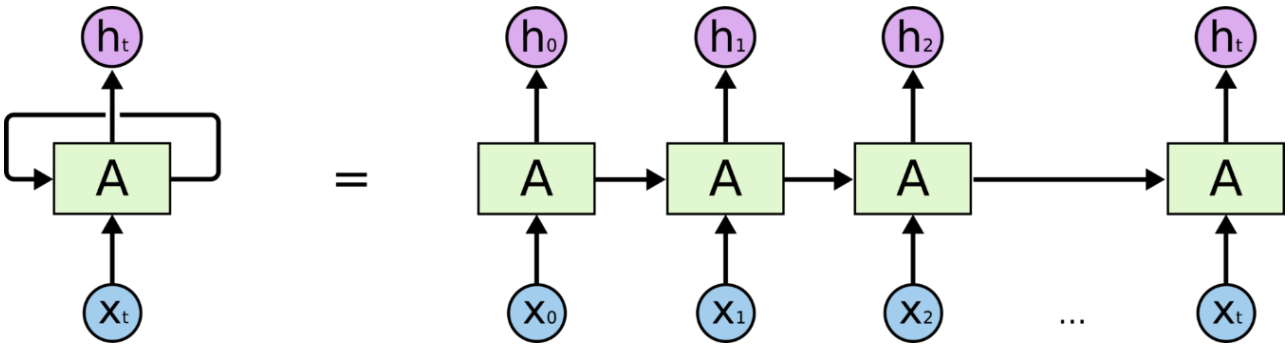

Рис. 4: рекуррентная нейронная сеть в развертке

На каждой итерации внутренний слой нейронов получает на вход новый член последовательности x_t и информацию о предыдущем состоянии внутреннего слоя A, на основании чего генерирует новый член последовательности h_t. Таким образом в сети реализуется «память», что принципиально меняет характер ее работы и позволяет анализировать любые последовательности данных, в которых важно, в каком порядке идут значения.

12 Приложение Б

LSTM

Любая рекуррентная нейронная сеть имеет форму цепочки повторяющихся модулей нейронной сети (см. рис. 5). В обычной РНС структура одного такого модуля очень проста, например, он может представлять собой один слой с

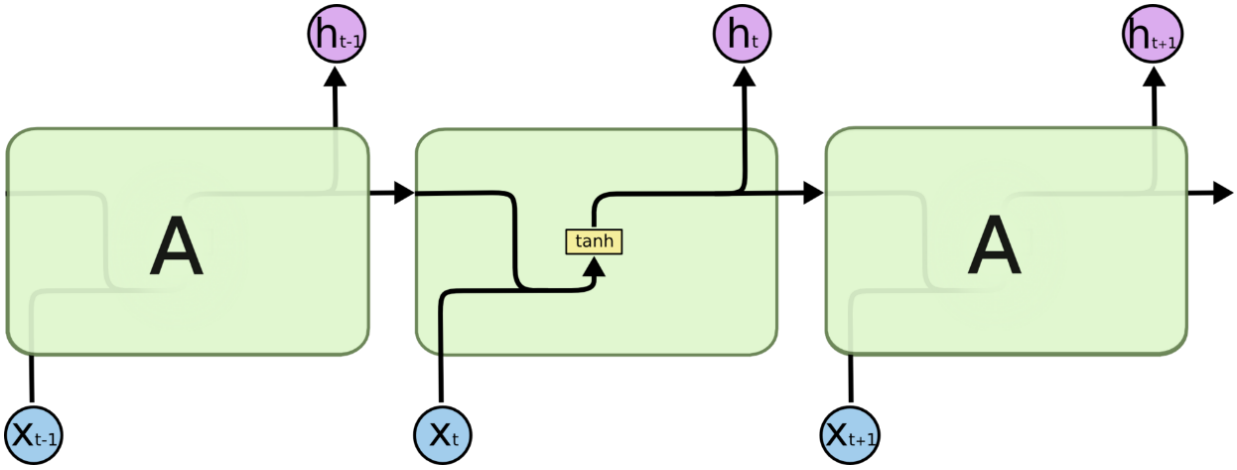

Рис. 5: слой рекуррентной нейронной сети

функцией активации tanh (гиперболический тангенс).

Структура LSTM также напоминает цепочку, но модули выглядят иначе (рис. 6). Вместо одного слоя нейронной сети они содержат целых четыре, и эти слои взаимодействуют особенным образом.

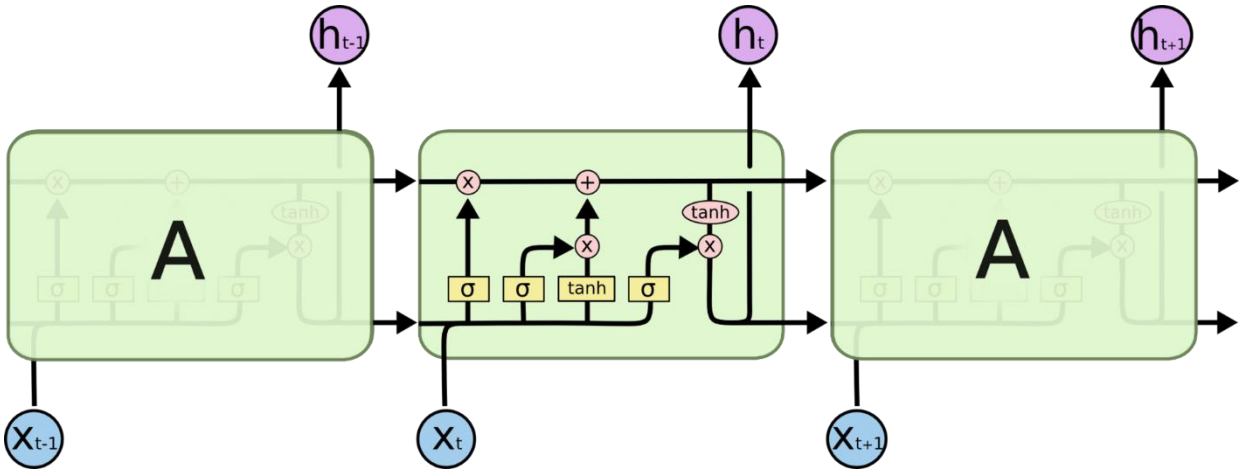

Рис. 6: слой LSTM

На рисунке 7 показаны специальные обозначения, используемые на схемах (слева направо: слой нейронной сети, поточечная операция, векторные перенос, объединение, копирование).

Ключевой компонент LSTM – это состояние ячейки (cell state) – горизонтальная линия, проходящая по верхней части схемы (рис. 8). Состояние ячейки проходит напрямую через всю цепочку, участвуя лишь в нескольких линейных преобразованиях. Информация может легко течь по ней, не подвергаясь изменениям.

Тем не менее, LSTM может удалять информацию из состояния ячейки; этот процесс регулируется структурами, называемыми фильтрами (gates). Фильтры позволяют пропускать информацию на основании некоторых условий. Они

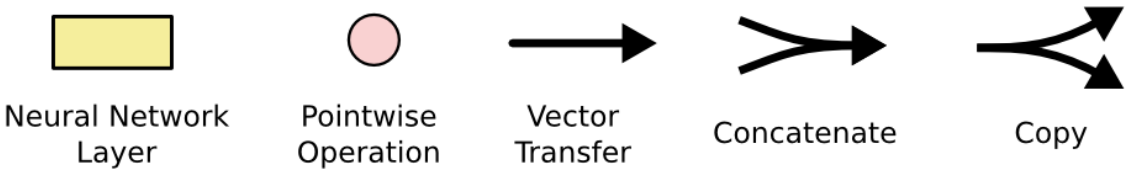

Рис. 7: условные обозначения

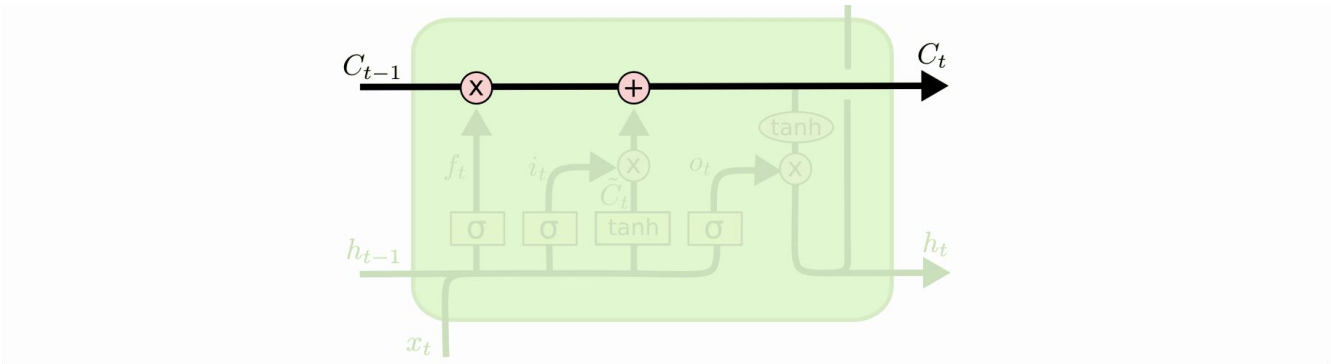

Рис. 8: состояние ячейки

состоят из слоя сигмоидальной нейронной сети и операции поточечного умножения.

Сигмоидальный слой возвращает числа от нуля до единицы, которые обозначают, какую долю каждого блока информации следует пропустить дальше по сети. Ноль в данном случае означает “не пропускать ничего”, единица – “пропустить все”.

В LSTM три таких фильтра, позволяющих защищать и контролировать состояние ячейки.