

Министерство науки и высшего образования Российской Федерации
Федеральное государственное бюджетное образовательное учреждение
Высшего образования
«Северо-Осетинский государственный университет
имени Коста Левановича Хетагурова»

Дипломная работа
Seq2seq подход для задач Машинного Перевода

Выполнил:
Студент 4 курса направления:
«Прикладная математика и информатика»
Гамосов Станислав Станиславович _____

Научный руководитель:
Кандидат физико-математических наук:
Басеева Елена Казбековна _____

Консультант
Старший преподаватель:
Макаренко Мария Дмитриевна _____

Владикавказ 2022

Содержание

1	Введение	2
2	Рекуррентные сети	3
2.1	RNN - Recurrent Neural Network	3
2.2	LSTM - Long Short-Term Memory	8
2.3	GRU - Gated Recurrent Unit	9
3	Формализация задачи машинного перевода	10
4	Структура Encoder-Decoder	11

1 Введение

Seq2seq - это семейство подходов машинного обучения, используемых для обработки естественного языка. Основные задачи для которого используются данные методы: нейронный перевод, субтитры к изображениям, разговорные модели и обобщение текста.

Первоначальный алгоритм, который в процессе породил целое семейство методов, был разработан *Google* для использования в машинном переводе. Как уже можно заметить за последнюю пару лет коммерческие системы стали удивительно хороши в переводе - посмотрите, например, *Google Translate*, *Яндекс-переводчик*, переводчик *DeepL*, переводчик *Bing Microsoft*.

Так же **Seq2seq** технология несет в себе огромный потенциал, помимо привычного машинного перевода между естественными языками, вполне реализуем перевод между языками программирования (*Facebook AI "Глубокое обучение переводу между языками программирования"*). Поэтому возможности применений такого рода подходов довольно велики. В связи с этим под машинным переводом будет подразумеваться любая задача **Seq2seq**, если точнее, то перевод между последовательностями любой природы.

2 Рекуррентные сети

2.1 RNN - Recurrent Neural Network

Одно из важных отличий RNN от обычных нейронных сетей это понятие времени. Под ним подразумевается последовательность входных данных x^t , которые поступают на вход, и их выходные результаты y^t , которые генерируются в дискретной последовательности временных шагов, индексированных t . Получаемые последовательности могут быть конечной длины или бесконечно счетными. Когда они конечны, так называемый отрезок времени примет вид $(\overline{1}, \overline{T})$.

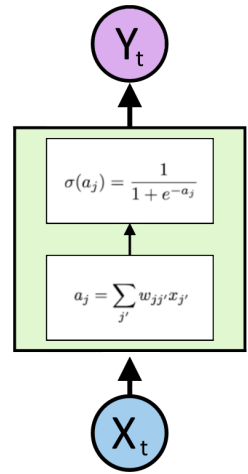
Таким образом, входную последовательность можно обозначить $x^t = (x^1, x^2, x^3, \dots, x^T)$, а выходную последовательность как $y^t = (y^1, y^2, y^3, \dots, y^T)$

Рекуррентные нейронная сеть как и обычные нейронные сети представляют из себя граф состоящий из набора искусственных нейронов, обычно называемых **узлами**, и набора направленных взвешенных ребер между ними. Каждый нейрон j связан **функцией активации** σ_j .

Вес - это связь между нейронами, которая несет в себе значение, которое характеризует "важность", придаваемая значению сигнала, проходящего через данное ребро или синапс. Для каждого ребра от узла j' до j присутствует вес $w_{jj'}$. Значение v_j каждого нейрона вычисляется путем применения его функции активации к взвешенной сумме его входных данных:

$$v_j = l_j \left(\sum_{j'} w_{jj'} \cdot v_{j'} \right)$$

Для удобства обозначим $a_j = \sum_{j'} (w_{jj'} \cdot v_{j'})$ и назовём **текущей активацией**. **Функция активации** $\sigma(z)$ является абстракцией, представляющей скорость возбуждения нейрона. Обычно в качестве функций активации применяют:



Функция Хевисайда	$H(x) = \begin{cases} 0, & x < 0 \\ 1, & x \geq 0 \end{cases}$
Сигмоида	$\sigma(x) = \frac{1}{1+e^{-x}}$
Гиперболический тангенс	$\tanh(x) = \frac{e^x - e^{-x}}{e^x + e^{-x}}$
Линейный выпрямитель	$ReLU(x) = \begin{cases} 0, & x < 0 \\ x, & x \geq 0 \end{cases}$

Рекуррентные Нейронные Сети (Resicrrrent Neural Network - RNN) - это класс сетей с циклами, которые хорошо подходят для обработки последовательностей. Мысли обладают неким постоянством и напрямую зависят от прошлых умозаключений. Традиционные нейронные сети на такое не способны, и это, очевидно, серьезный изъян.

Допустим перед нами стоит задача научить сеть определять эмоциональный окрас предложения, и подаём в сеть одно слово за другим. Желательно, чтобы сеть "помнила" уже переданные слова. Уже здесь возникает проблема обычных нейронных сетей, как же "запоминать" контекст? Если мы хотим, чтобы сеть переводила предложение с одного языка на другой, то тоже было бы не плохо учитывать начало, середину и конец предложение при переводе. В таких случаях именно рекуррентные нейронные сети призваны решить такие проблемы.

В более общем плане сети RNN могут работать с последовательностями (sequence) произвольной длины, а не с входными данными фиксированного размера. Это свойство как раз таки очень важно в контексте обработки естественных языков. Также хороший пример, потому что нейронная сеть должна учитывать контекст, предоставляемый существующим предложением, чтобы завершить его.

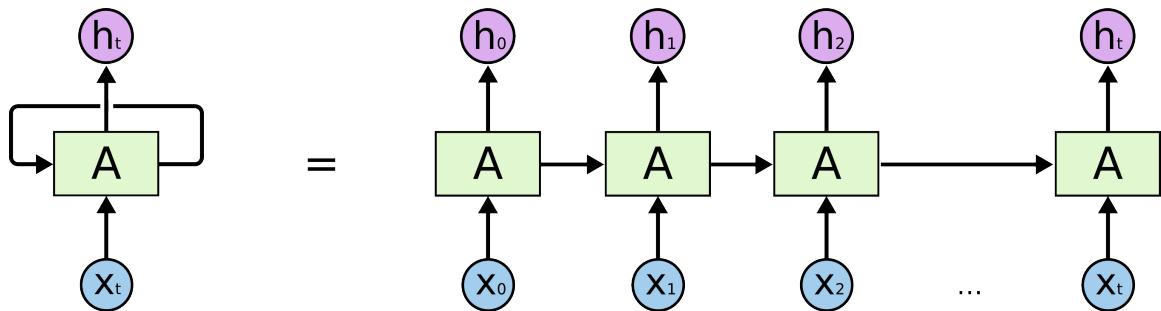


Рис. 1: Развернутая рекуррентная нейронная сеть

Такая "цепная" сущность показывает, что рекуррентные нейронные сети по природе своей тесно связаны с последовательностями. Естественно использовать такую архитектуру для работы с этим типом данных.

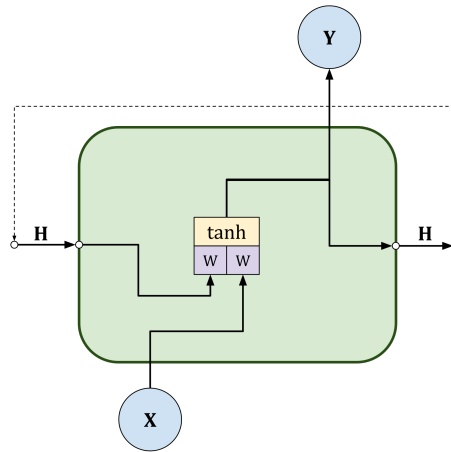
Кроме выходного вектора, где мы будем получать ответ, сеть должна иметь еще и некоторый вектор или векторы v , которых описывает текущее внутреннее состояние сети, т.е. в нем содержатся воспоминания о всех уже рассмотренных сетью элементах. Более формально это выглядит так.

Пусть у нас есть входная последовательность $(x_1, x_2, x_3, \dots, x_n)$, данную последовательность стоит преобразовать:

$$h^{(t)} = \sigma(w^{(hx)} \cdot x^{(t)} + w^{(hh)} \cdot h^{(t-1)} + b_h)$$

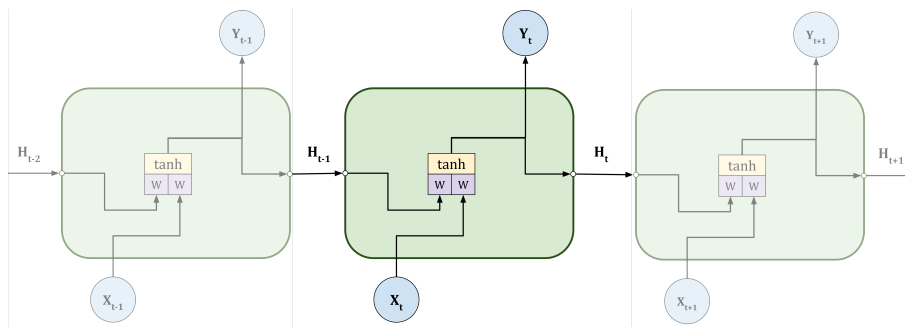
$$y^{(t)} = w^{(yh)} \cdot h^{(t)} + b_y$$

При этом кроме выхода $y^{(t)}$, мы имеем еще вектор $h^{(t)}$, описывающий текущее состояние. Таким образом сеть состоит из ячеек вида:



Которые собираются в последовательность, передавая внутреннее состояние из ячейки в следующую за ней по времени. Отметим, что веса при этом у всех ячеек одни и те же:

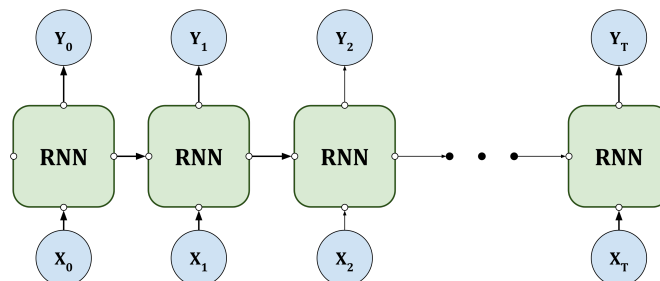
Ещё один вариант, когда сеть преобразует последовательность в последовательность, может быть вот таким:



Можно выделить три варианта конфигурации такой сети.

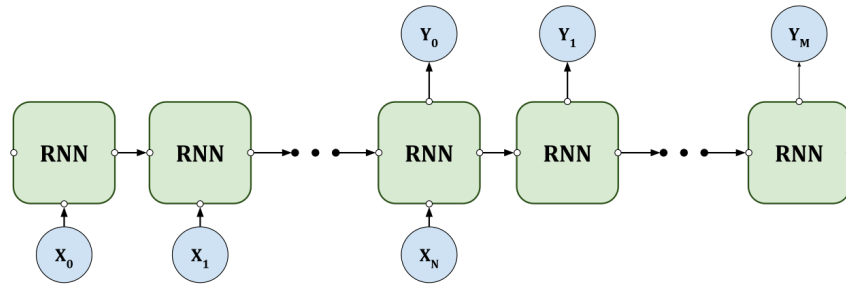
Sequence to sequence

Эта конфигурация применяется, когда мы хотим при помощи сети преобразовать последовательность в последовательность. Обычно это генеративная модель, например, для текстов, на каждом шаге в этом случае сеть выдаёт новую букву (вернее вектор вероятностей для букв алфавита).



Ещё один вариант, когда сеть преобразует последовательность в последовательность, может быть вот таким:

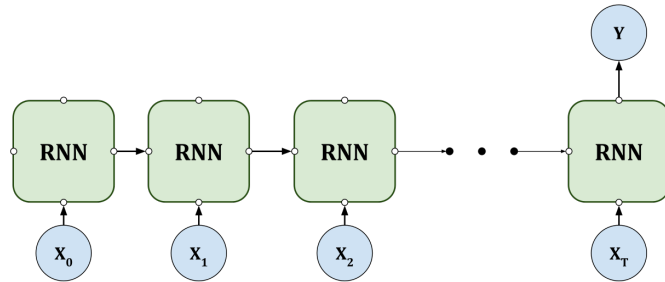
На самом деле это сильно похоже на вариант encoder-decoder, когда входная последовательность кодируется (закодированные данные оказыва-



ются во внутреннем состоянии сети), а затем декодируется. Например, такая схема используется для переводов текстов.

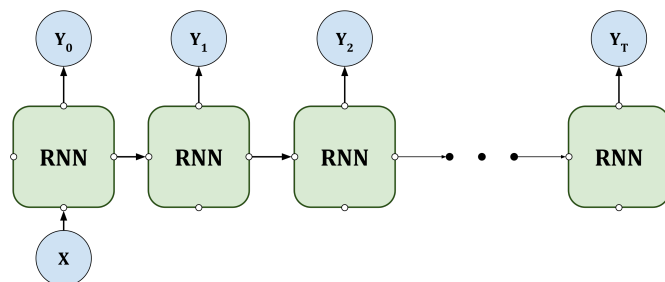
Sequence to one

Эта конфигурация применима для решения задач классификации, например, текстов или видео, когда последовательность слов (или изображений) подаются на вход сети, а в качестве выхода получаем один вектор вероятностей для классов.



One to sequence

Ситуация обратная предыдущей, на вход приходит один элемент, а на выходе получаем целую последовательность. Например, таким образом можно генерировать текстовые подписи к изображению подаваемому на вход.



2.2 LSTM - Long Short-Term Memory

2.3 GRU - Gated Recurrent Unit

3 Формализация задачи машинного перевода

Формально в задаче машинного перевода у нас есть входная последовательность x_1, x_2, \dots, x_m и последовательность вывода y_1, y_2, \dots, y_n , само собой длинна данных последовательностей может отличаться. Саму процедуру *перевода* можно рассматривать как нахождение искомой последовательности, которая является наиболее вероятной с учетом входных данных. Формально искомая последовательность, которая максимизирует условную вероятность $p(y|x) : y' = \operatorname{argmax}[p(y|x)]$.

Когда человеку известны уже два языка с которыми он работает, то уже при переводе можно сказать насколько хорошо справилась модель, является ли перевод естественным и насколько он приятен на слух. Однако такой вид анализа неприемлем для машины, поэтому нам стоит проанализировать уже имеющуюся функцию $p(y|x, \theta)$ с неким параметром θ , а затем найти его argmax для $y' = \operatorname{argmax}_y[p(y|x, \theta)]$.

Прежде чем перейти к самой задаче перевода, нужно ответить на 3 вопроса:

- **Моделирование:** Как работает модель для $p(y|x, \theta)$?
- **Обучение:** Как найти параметр θ ?
- **Вывод:** Как понять, что текущий y лучший?

4 Структура Encoder-Decoder

Наиболее распространенная модель **Sequence-to-sequence** (**seq2seq**) является модель **Encoder-Decoder**, в которой обычно используют **рекуррентную нейронную сеть** (**RNN**) для кодирования исходной последовательности в один вектор.

На самом деле полученный вектор можно представить как набор образов сущностей с образами взаимоотношений между ними. Этот вектор затем декодируется вторым **RNN**, который учится выводить выходное предложение, генерируя его по одному слову за раз.