Формализованная задача: отделение контента от стиля №1 с последующим совмещением контента со стилем №2.

Данные: блок (~ 5000 строк), предложения из статей wiki (~6000).

Вариант 1. Каталог: /../__test/simple_lm/
Простой вариант. У нас есть тексты двух стилей - для каждого тренируем модель seq2seq. Далее берем предложение стиля №1 прогоняем через энкодер, обученный на текстах данного стиля. Берем вектор сгенерированный последним и подаем на вход декодеру, обученному на стиле №2.

Почему должно работать: так как стиль автора остается неизменным, то модель сохраняет только содержание - то, что меняется от предложения к предложению. То есть, энкодер должен сохранять только контент.

Почему не работает: чем меньше похожи словари двух стилей, тем сложнее. У нас они почти не пересекаются.

Вариант 2. Каталог /../__test/linguistic_style_transfer/scripts/
Существует группа работ, которые решают близкую задачу на данных отзывов с сервиса Yelp. Необходимо трансформировать отзыв из позитивного в негативный с сохранением контекста и наоборот:

1. Multi-decoder and style-embedding models. Задача решается двумя моделями следующим образом: 1. Encoder -> output_vector (z) -> Adversarial_network. Последняя состоит из двух сетей с собственным набором параметров. Первая классифицирует стиль на основе z, вторая делает классификатор неспособным идентифицировать стиль, увеличивая энтропию. В результате мы получаем z, в котором нет контента, который подаются на несколько декодеров. 2. Encoder -> output_vector (z) -> Adversarial_network -> z (без контекста) конкатенируется с style_embedding (матрица параметров размером количество стилей x размерность вектора) -> decoder (1) Ссылка: https://arxiv.org/abs/1711.06861
2. CSAIL team. Суть: AE или VAE. Есть вектор скрытых переменных z. Специальная полносвязная сеть (discriminator neural network) на основе этого вектора пытается предсказать стиль, если ей это удается, то значит z содержит стилевые фичи - это плохо. Поэтому с помощью совместной оптимизации двух сетей нужно добиться того, чтобы у дискриминатора лосс был высоким, но reconstruction error низким - это возможно только в случае отсутствия стиля в z. Генерация фраз осуществляется отдельным декодером. Ссылка: https://papers.nips.cc/paper/7259-style-transfer-from-non-parallel-text-by-cross-alignment.pdf

3. Выбранный подход для экспериментов. Ссылка: https://arxiv.org/abs/1808.04339 Рассмотрим подробнее его особенности:

x -> Encoder (BiGRU) -> z -> m_style = Dropout(Dense(8)) -> sampled_style_embedding (based on m_s & sigma_s)
-> sigma_style = Dropout(Dense(8))

-> m_content = Dropout(Dense(128)) -> sampled_content_embedding (based on m_c & sigma_c)
-> sigma_content = Dropout(Dense(128))

-> (1) Dense(256)([sampled_style_embedding, sampled_content_embedding]) -> Decoder(GRU) -> RL
-> (2) Dropout(Dense(m_style, units=labels_number, softmax)) -> style_multitask_loss(softmax_crossentropy) (SMT)
-> (3) Dropout(Dense(m_content, units=BOW_size, softmax)) -> style_multitask_loss() (CMT)
-> (4) Dropout(Dense(m_content), 128) -> Dense(количество классов=2, softmax) -> style_adversarial_entropy (SE)
-> style_adversarial_loss (cross_entropy)
-> (5) Dropout(Dense(m_style), units=bow_size) -> Dense(units=bow_size, softmax) -> content_adversarial_entropy (CE)
-> content_adversarial_loss (cross_entropy)
-> (6) Dropout(Dense([m_style, m_content], units=num_labels, softmax)) -> style_overall_prediction_loss(softmax_cross_entropy)

В данной модели есть несколько loss function:

1. Reconstruction loss. (RL) Обычно используют расстояние Кульбака-Лейблера (KL) с нормальным распределением. Но, автор использует ELBO (the evidence lower-bound), что позволяет тренировать декодер как generative network и использовать его независимо от остальной модели.
2. Auxiliary loss. Multitask loss (style). Цель: в векторе стиля есть информация о стиле.
3. Auxiliary loss. Multitask loss (content). Цель: в векторе контента есть информация о контенте.

4. Adversarial style loss. Идея: adversarial discriminator предсказывает стиль через softmax.

It is trained by minimizing the following objective, using a cross-entropy loss.

$$J_{\text{dis}}(\boldsymbol{\theta}_{\text{dis}}) = -\mathbb{E}[\log p(y'|\boldsymbol{c}; \boldsymbol{\theta}_{\text{dis}})] \quad (6)$$

where y' is the true label distribution.

При этом, параметров энкодера здесь нет, так как мы хотим, чтобы он научился исключать стиль из z, и тогда дискриминатор ничего не найдет: Then the autoencoder is trained to learn a content vector space from which its adversary cannot predict style information. С этой целью параметры энкодера обновляются следующим образом:

$$J_{\text{adv}}(\boldsymbol{\theta}_E) = \mathcal{H}(y|\boldsymbol{c}; \boldsymbol{\theta}_{\text{dis}}) \quad (7)$$

where $\mathcal{H}(p) = -\sum_{i \in \text{labels}} p_i \log p_i$ is the entropy and y is

Мы рассчитываем энтропию предсказаний дискриминатора и пытаемся ее максимизировать. Мотивация заключается в увеличении неопределенности дискриминационного классификатора в отношении прогнозирования стиля из вектора контента.

5. Adversarial content loss. Повторить тоже самое, но для контента. Вводится понятие контента - слова из оригинального предложения - стоп-слова, слова, однозначно относящиеся к определенному стилю. Дискриминатор предсказывает вероятность появления каждого слова словаря в исходном предложении. Мотивация: хотим, чтобы энкодер выучил представления стиля, в котором bag-of-words discriminator не смог бы предсказать большинство исходных слов.

6. Auxiliary loss. Style Используется для того, чтобы убедиться в том, что мы ничего не потеряли в результате всего вышеперечисленного.

7. Composite loss. Комбинируем некоторые из вышеперечисленных лоссов. Но минимизируем в следующем порядке: сначала 2 adversarial, потом composite.

Генерация текста:

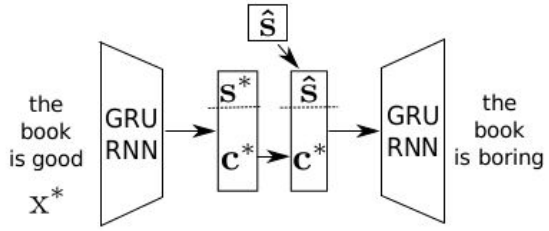


Figure 2: Model Generation Overview

Замена исходного вектора стиля на усредненный вектор нужного стиля.

Метрики:

1. Style Transfer. Тренируется отдельная сеть для классификации стиля. Считается acc или f1 - сколько из сгенерированных предложений могут быть отнесены к целевому стилю.
2. Content Preservation.Обучаются w2v модели. We compute a sentence embedding by min, max, and average pooling its constituent word embeddings. Then, the cosine similarity between the source and generated sentence embedding is computed to evaluate how close they are in meaning. Из слова исключаются стоп-слова.
3. Word Overlap. Simpler unigram overlap metric.
4. Language Fluency. Тренируем 3-грамную языковую модель со сглаживанием Knerser-Ney. Потом считаем log-likelihood скор.

Эксперименты.

Первая проблема была связана с Adversarial content loss. На схеме видно, что у нас получается матрица параметров [vocub x vocub]: *Dropout(Dense(m_style), units=bow_size) -> Dense(units=bow_size, softmax).* На данных Yelp это не проблема, так как словарь = 10k. Но на корпусе стихов и wiki это становится проблемой, так как слова, выходящие за пределы диапазона словаря, помечаются специальным токеном типа <unk>, в результате при генерации получаются бессмысленные наборы слов. Плюс, необходимость данного компонента вызывает сомнения, так как без него все ok.

Objectives	Transfer Strength	Content Preservation	Word Overlap	Language Fluency
J_{rec}	0.144	0.915	0.329	-14.28
J_{rec}, J_{adv}	0.727	0.880	0.204	-14.16
J_{rec}, J_{mult}	0.789	0.898	0.259	-14.56
J_{rec}, J_{badv}	0.168	0.915	0.328	-14.45
J_{rec}, J_{adv}, J_{mult}	0.890	0.882	0.211	-14.41
J_{rec}, J_{adv}, J_{badv}	0.749	0.883	0.202	-14.36
J_{rec}, J_{mult}, J_{badv}	0.783	0.896	0.257	-14.34
J_{rec}, J_{adv}, J_{mult}, J_{badv}	0.885	0.878	0.197	-14.05

Для экспериментов он был отключен.

Каталог: /../__test/linguistic_style_transfer/scripts/output/20180904181854-inference/

Вторая проблема: низкое значение style transfer при высоком content preservation. Это означает, что модель при генерации максимально воспроизводит стиль исходной фразы.

Примеры “интересных” сгенерированных предложений:

арочные плотины давление от масс воды передают на берега ущелья реже на искусственные устои в силу этого такие плотины чаще	*арочные плотины может быть изолированные от воды смотрел на берега ущелья реже на красавице не страшна герою пока не*

вход в дом находится в нише перед дверью поставлены две скамьи по сторонам от входа два торговых помещения росписи	*вход на улице стоял дом и на усталый конь быстрее скачет к цели в чужом селе мерцают огоньки по*
церковь св марии была построена в начале xiv века гг и выполнена в традиционном для этой местности стиле	*и вдруг в небывалых славах принесла нам вздохи курений*
при этом зависимость доходов населения от увеличивается сокращения доли удельный вес социальных выплат в структуре доходов россиян во	*и не помню суровых чудес на заре голубые химеры смотрят в зеркале ярких небес*
сегодня точно не известно как проходили первые епископов но можно допустить что первые епископы выбирались апостолами и их ближайшими	*сегодня точно не надо как в гробу стеклянном и не мертва так и в этом комнатном теплом углу поглядим*

Возможная проблема: posterior collapse (хотя в этом случае decoder должен игнорировать z, но дополнительные метрики на основе KL остаются небольшими), не может выйти из локального минимума (возможно, поможет настройка гиперпараметров сети (например, размер style_vector, который = 8, регуляризация)), разность словарей (использовать НКРЯ).

Вариант 3. Использование Markov chain.

Источники:

- http://amsdottorato.unibo.it/5685/1/barbieri_gabriele_tesi.pdf
- http://amsdottorato.unibo.it/5685/1/barbieri_gabriele_tesi.pdf

Описание возможного подхода: использование цепей Маркова, но с ограничениями, которые позволяют контролировать генерацию текста: ключевые слова, типы рифм.

У нас есть корпус, состоящий из нескольких предложений: *Clay loves Mary, Mary loves Clay, Clay loves Mary today, Mary loves Paul today.*

Матрица переходов:

$$T_{ex} = \begin{pmatrix} 0 & 0 & 0 & 0.67 & 0.33 \\ 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 1 \\ 0.5 & 0.25 & 0.25 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 \end{pmatrix}$$

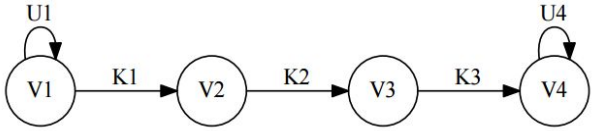
Вектор априорных вероятностей: [1/7, 3/14, 1/14, 2/7, 1/7]

M = M ary, C = Clay, P = P aul, L = loves, T = today.

Пример: предложение *Clay loves Mary loves* будет иметь вероятность: p(CLML) = 1 / 14

Вводим **control constraint:** первое слово должно рифмоваться со словом *today,* последнее слово должны быть *today.*
Формально: *U1* = {Clay, today} , *U4* = {today}.

Constraint graph.



Далее для применения указанных ограничений мы выполняем следующие шаги:

1. The first transformation exploits the induced CSP (имеется в виду binary-sequential CSP) to filter out state transitions that are explicitly or implicitly forbidden by the constraints. This is achieved by replacing the corresponding transition probabilities by zeros in the initial transition matrices. A side-effect is that the transition matrices are not stochastic anymore (rows do not sum up to 1 any longer).
2. The second transformation consists in renormalizing those matrices to obtain a proper (non-homogeneous) Markov model.

В результате у нас будет два предложения, которые могут быть сгенерированы данными матрицами: *Clay loves Mary today, Clay loves Paul today*.

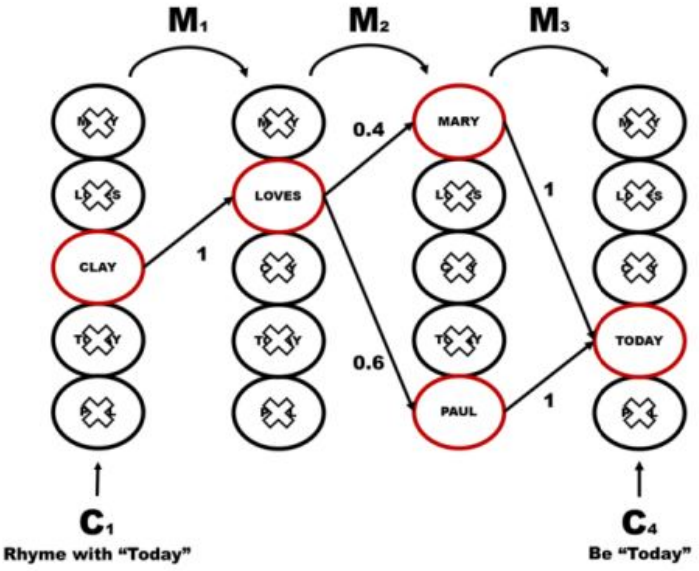


Figure 2. A constrained Markov process $\tilde{M}_{ex}$ that generates verses composed of 4 words and rhymes with the word *yesterday*. $\tilde{M}_{ex}$ and M_{ex} have the same probability distribution. M_1, M_2 and M_3 represent the Markov constraints, C_1 represents the control constraint "rhyme with today", C_4 represents the control constraint "be today". The arrow labels indicate the transition probabilities.

Возможные ограничения:

1. Рифма. У нас есть целевое слово, в соответствии с фонетическим произношением, мы выберем слова, которые рифмуются с целевым словом. Мы используем фонетическую транскрипцию, основанную на правилах, чтобы на основе совпадения суффиксов определять рифмуются слова или нет.
2. Метр стиха. Это положение ударных слогов в стихе. Каждое слово должно иметь свой ритмический тег: 01 - второй ударный слог. Метр стиха составлен ритмическими тегами его слов. Например: [101, 1, 1, 10, 1, 1, 1, 01].
3. POS. Пример шаблона: [PRP, VBD, IN, PRP, RB]
4. Семантические. Для любого слова в корпусе мы должны найти n близких к нему по значению. Воспользуемся w2v.

Реализовать данный перспективный подход не хватило времени.