Автоматическая суммаризация текста

Попов Артём, осень 2022

Natural Language Processing

Задача суммаризации текста (text summarization)

Суммаризация – построение по одному или нескольким документам краткого текста (абстракт, summary) наиболее полно передающего их содержание.

Основные типы задач суммаризации:

- one-document – на входе один документ
- multi-document – на входе несколько документов
- topic – на входе кластер документов, у каждого документа есть вес

Два основных подхода к суммаризации:

- extractive – выбор предложений из исходного текста
- abstractive – генерация абстракта

Пример суммаризации

Abstract

We present a method to produce abstractive summaries of long documents that exceed several thousand words via neural abstractive summarization. We perform a simple extractive step before generating a summary, which is then used to condition the transformer language model on relevant information before being tasked with generating a summary. We show that this extractive step significantly improves summarization results. We also show that this approach produces more abstractive summaries compared to prior work that employs a copy mechanism while still achieving higher rouge scores. *Note: The abstract above was not written by the authors, it was generated by one of the models presented in this paper based on an earlier draft of this paper.*

S.Subramanian et al (2019); On Extractive and Abstractive Neural Document Summarization with Transformer Language Models

Зачем нужна суммаризация?

Практические приложения суммаризации:

- выделение полезной информации из научных статей
- составлений дайджестов научных статей
- генерация сниппетов/заголовков новостей

Наблюдение. Чтобы сжать текст без потерь информации, необходимо полностью осознать его смысл.
Таким образом, суммаризация – сложная и показательная задача для всей области NLP.

Метрика ROUGE (Recall-Oriented Understudy for Gisting Evaluation)

Средняя или максимальная доли n-грамм из правильных абстрактов, вошедших в суммаризацию s:

$$ROUGE_n(s) = \frac{\sum_{r \in \mathrm{R}} \sum_w \mathbb{I}[w \in s] \mathbb{I}[w \in r]}{\sum_{r \in \mathrm{R}} \sum_w \mathbb{I}[w \in r]}$$

$$ROUGE_{n\,multi}(s) = \max_{r \in R} \frac{\sum_w \mathbb{I}[w \in s] \mathbb{I}[w \in r]}{\sum_w \mathbb{I}[w \in r]}$$

- $r \in R$ – множество абстрактов, написанных людьми
- s – абстракт, построенный системой
- чем больше, тем лучше – для всех метрик семейства ROUGE

Chin-Yew Lin (2004); ROUGE: A package for automatic evaluation of summaries

Оценивание суммаризации с точки зрения теории информации

Суммаризация — задача семантического сжатия информации.

- Энтропия – мера неопределённости распределения

$$H(p) = -\sum_i p_i \log p_i$$

- Кросс-энтропия – мера неопределённости, возникающая при наблюдении p при ожидании q

$$CE(p, q) = -\sum_i p_i \log(q_i)$$

- KL-дивергенция – потери информации при использовании p вместо q

$$KL(p, q) = \sum_i p_i \log(p_i/q_i) = CE(p, q) - H(p)$$

Оценивание суммаризации с точки зрения теории информации

Пусть текст d, абстракт s, фоновая информация b определяются распределением над семантическими элементами:

- частота токенов в документе $\frac{n_{wd}}{\sum_{w \in d} n_{wd}}$
- расстояния от документа до центров кластеров $\frac{\rho(d,c)}{\sum_c \rho(d,c)}$

Качество абстракта можно оценить по формуле:

$$Q(s,d,b) = H(s) - \alpha CE(s,d) + \beta(s,b)$$

Peyrard (ACL 2019); A Simple Theoretical Model of Importance for Summarization

Оценивание суммаризации с точки зрения теории информации

$$Q(s, d, b) = H(s) - \alpha CE(s, d) + \beta CE(s, b)$$

- $H(s)$ должно быть большим, так как абстракт должен содержать большое количество информации
- $CE(s, b)$ должно быть большим, абстракт не должен содержать общей информации
- $CE(s, d)$ должно быть маленьким, абстракт должен быть похож на исходный документ

Основные этапы extractive суммаризации

1. Построение представления предложений внутри документа
2. Ранжирование предложений
3. Отбор предложений для абстракта
4. Сортировка/группировка предложений в абстракте

Мы можем как и обучать модель ранжирования (при наличии данных), так и использовать обучение без учителя.

Построение представления предложений

Агрегация эмбеддингов токенов:

- усреднение или сумма
- взвешенное усреднение или сумма
- max/min pooling
- иерархический pooling
- конкатенация по нескольким способам

Эмбеддинги могут быть таблицей векторов для заданных токенов (word2vec) или быть контекстно-зависимыми, получаемыми из сложной модели (BERT).

Обучаемые представления предложений

В обучение BERT не закладывается последующая агрегация внутренних представлений для эмбеддинга предложения!

Мы можем обучать представления на задачах, связанных с близостью предложений.

Основная задача для большинства моделей – SNLI, определение связи двух предложений (contradiction, eintailment или neutral).

A black race car starts up in front of a crowd of people.	contradiction C C C C C	A man is driving down a lonely road.
A soccer game with multiple males playing.	entailment E E E E E	Some men are playing a sport.
A smiling costumed woman is holding an umbrella.	neutral N N E C N	A happy woman in a fairy costume holds an umbrella.

https://nlp.stanford.edu/projects/snli/

Infersent. Обучение

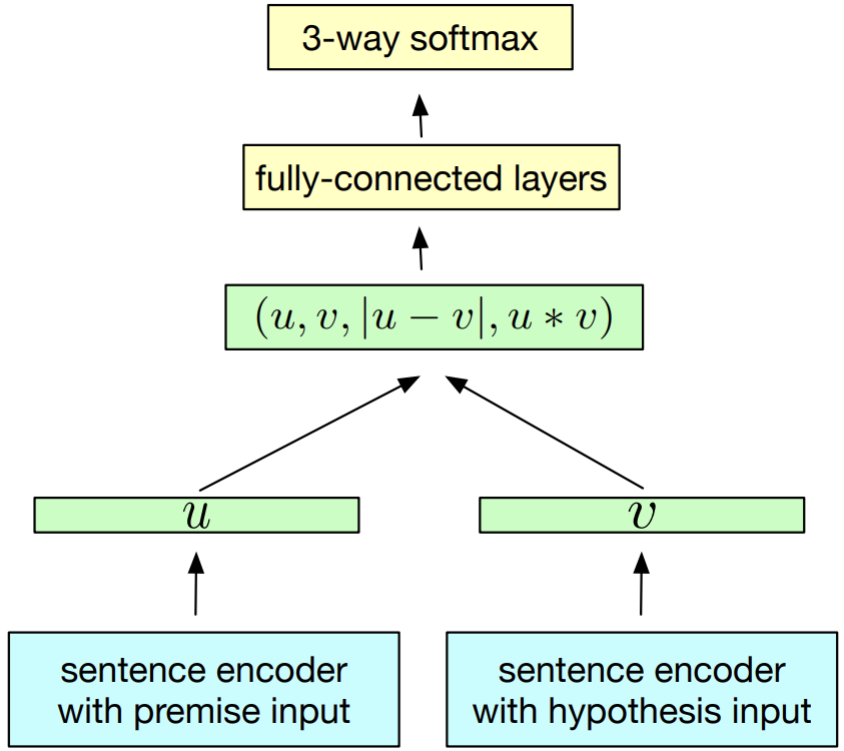
Вход: два предложения, каждое подаётся в энкодер – biLSTM

Эмбеддинг предложения – усреднение или max pooling скрытых состояний всех позиций.

Вектор признаков – конкатенация из:

- каждого из двух эмбеддингов
- поэлементной разности и произведения двух эмбеддингов

Вектор признаков пропускается через несколько полносвязных слоёв и подаётся в softmax.



Conneau et al (2018); Supervised Learning of Universal Sentence Representations from Natural Language Inference Data

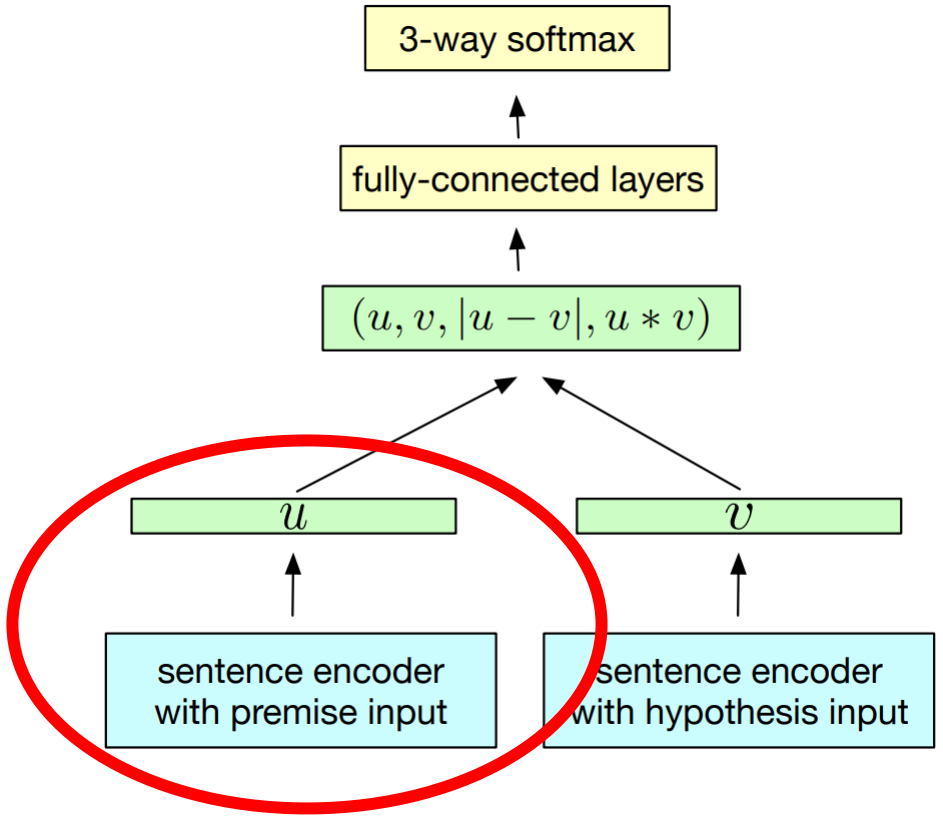
Infersent. Применение

На этапе применения из всей архитектуры используется только энкодер.

В качестве функции близости обычно используется косинусная близость.

Преимущества архитектуры:

- можем заранее построить индекс: эмбеддинги всех предложений
- на этапе применения нужно будет только вычислять близости между векторами



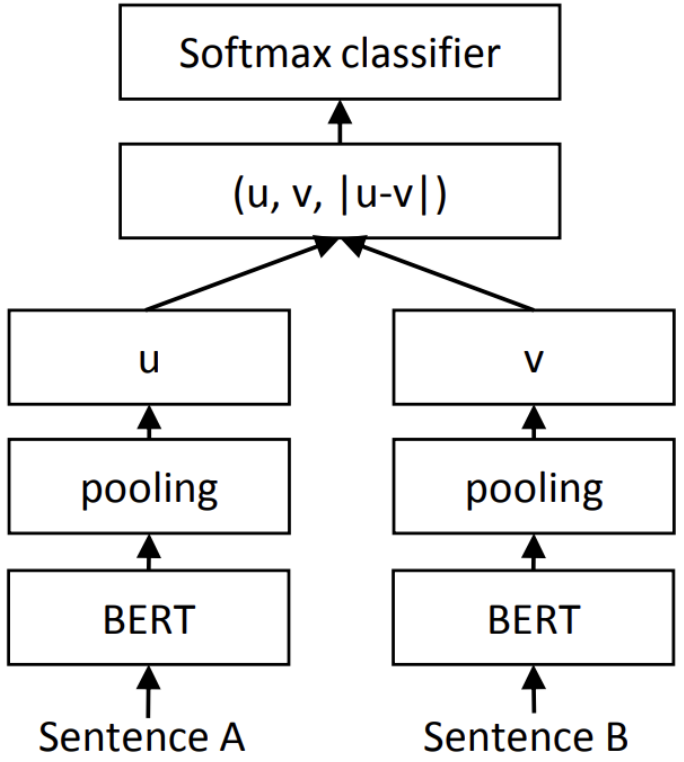
Сиамские нейросети на основе трансформеров

Universal Sentence Encoder (USE):

- вместо LSTM используем трансформер
- к классификации добавляют задачу восстановления следующего и предыдущего предложений

Sentence-BERT (sBERT):

- инициализируем энкодер моделью BERT, дообучаем под задачу SNLI



Cer et al (2018); Universal Sentence Encoder

Reimers et al (2019); Sentence-BERT: Sentence Embeddings using Siamese BERT-Networks

Ранжирование предложений: разметка

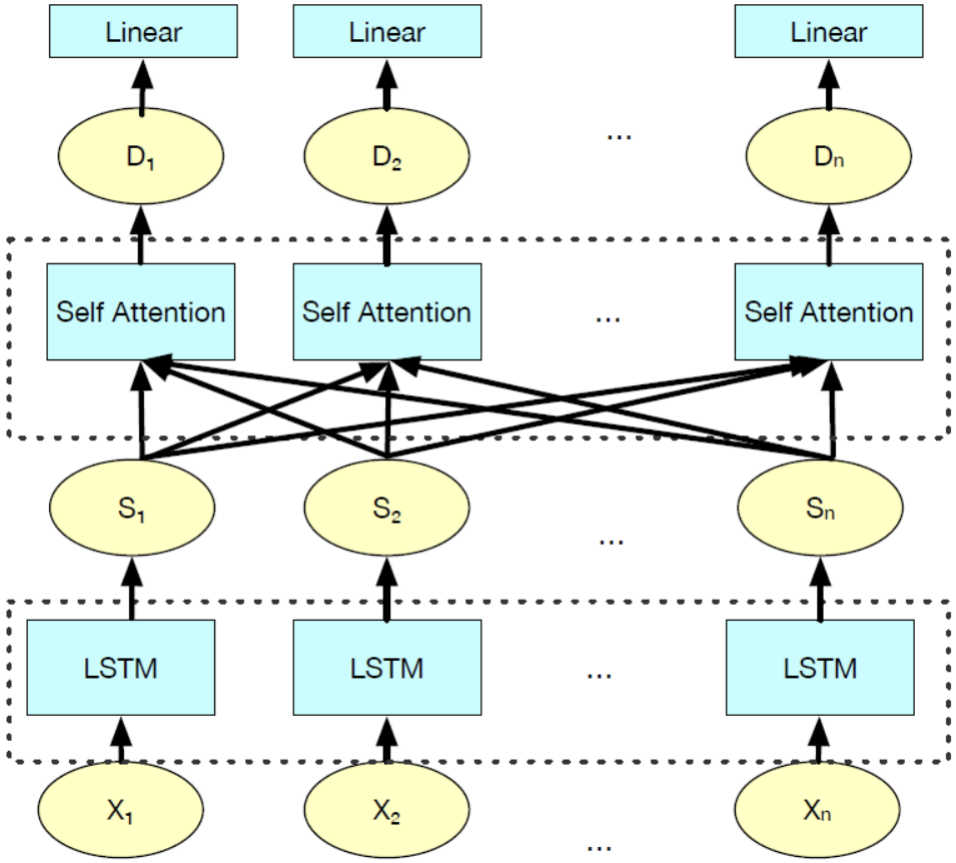
Extractive суммаризация — задача разметки предложений с классами «включать в абстракт» и «не включать».

Можно использовать нейросетевые модели для работы с последовательностями эмбеддингов предложений, вероятность “включения в абстракт” – релевантность для ранжирования.

Проблема: данных мало, а модели переноса обучения работают на уровне токенов, а не предложений.

Контекстно-зависимые эмбеддинги предложений

- Каждое предложение задаётся последовательностью слов
- Эмбеддинг предложения – LSTM по токенам предложения
- Несколько слоёв трансформера отвечают за учёт контекста
- На выходе сети решаем одну из синтетических задач
- После предобучения применяем fine-tuning для решения конечной задачи



[Wang et al (2019); Self-Supervised Learning for Contextualized Extractive Summarization]

Self-supervision задачи для предобучения сети

MASK: $p_m = 0.25$

- маскируем предложение
- необходимо выбрать нужное предложение из набора пропущенных

Replace: $p_r = 0.25$

- заменяем предложение на случайное из другого документа
- для каждого предложения предсказываем, было ли оно заменено

Switch: $p_s = 0.25$

- заменяем предложение на случайное из данного документа
- для каждого предложения предсказываем, было ли оно заменено

Ранжирование предложений: TextRank

TextRank – unsupervised алгоритм присваивания релевантности предложениям.

Пусть наш документ d – взвешенный полный граф, вершинами которого являются предложения s, а веса ребёр задаются близостью предложений $sim(s, s')$ согласно некоторой модели.

Предложение s тем релевантнее, чем:

- больше других предложений s' похожих на s
- чем важнее те предложения s', которые похожи на s
- чем меньше других предложений, на которые s' тоже похожи

Mihalcea et al (EMNLP 2004); TextRank: Bringing Order into Texts

TextRank: получение релевантностей

$TR(s)$ – релевантность предложения, вычисляется через итеративный процесс, имитирующий случайные блуждания.

$$TR_{new}(s) = (1-\delta)\,\frac{1}{|d|} + \delta \sum_{s' \in d} p(s|s')\,TR_{old}(s')$$

$$p(s|s') = \frac{sim(s', s)}{\sum_{c \in d} sim(s', c)}$$

где $\delta \in (0, 1)$ – вероятность продолжения блуждания, а в качестве sim можно использовать косинусную близость эмбеддингов предложений или WMD подобные алгоритмы

Выбор количества предложений

У нас есть предложения с присвоенными им релевантностями!

Как выбрать из них предложения для абстракта?

- выбрать N наиболее релевантных, где N – константа
- разделить предложения на категории (классификация, кластеризация), выбрать N из каждой категории
- ввести критерий “идеальности” абстракта, итеративно добавлять по одному предложению до тех пор пока критерий уменьшается
- обучить модель, оценивающую близость подвыборки предложений документу, выбрать наилучшую подвыборку

Zhong et al (ACL 2020); Extractive Summarization as Text Matching

Резюме по extractive суммаризации

- Можно делать как supervised, так и unsupervised
- На текущий момент используется в коммерческих проектах чаще чем abstractive, особенно для длинных текстов
- Простой выборки из предложений может быть недостаточно, дополнительно можно:
  - разбить предложения по категориям (например, results, methods, background)
  - расположить предложения в удобном для просмотра порядке (например, согласно их появлению в тексте)
  - предоставить набор найденных именованных сущностей в документе

Абстрактивная суммаризация

Хотим решать задачу при помощи генеративной модели:

- **вход** – исходный текст
- **выход** – абстракт

Можем использовать всё, что мы проходили до этого:

- преобразования последовательности, энкодер-декодер на основе RNN с механизмом внимания или трансформера
- языковое моделирование, декодер на основе RNN или трансформера
- BPE токенизация
- Авторегрессионная генерация текста и различные трюки: beamsearch, температура, topK и topP сэмплирование

Проблемы и особенности задачи

- Недопустимы никакие искажения информации, так как предполагается, что читатель абстракта может не читать исходный текст.
- Нужна большая обучающая выборка текст-абстракт.
- Исходный текст может иметь большую длину.
- Часто основной домен данных – научные публикации

Идея: хотим совместить abstractive и extractive подходы. Модель генерирует абстракт, но некоторые части копирует из исходного текста.

Кодировщик-декодировщик LSTM + внимание

Пересчёт скрытых состояний кодировщика по $x = (x_1, \dots, x_n)$:

$$h_i = GRU_{enc}(Emb(x_i), h_{i-1})$$

Вычисление весов внимания:

$$\alpha_{ij} = softmax_{i \in \{1,\dots,n\}}\left(sim\left(h_i, z_{j-1}\right)\right)$$

Вычисление вектора-контекста внимания:

$$c_j = \sum_{i=1}^{n} \alpha_{ij} h_i$$

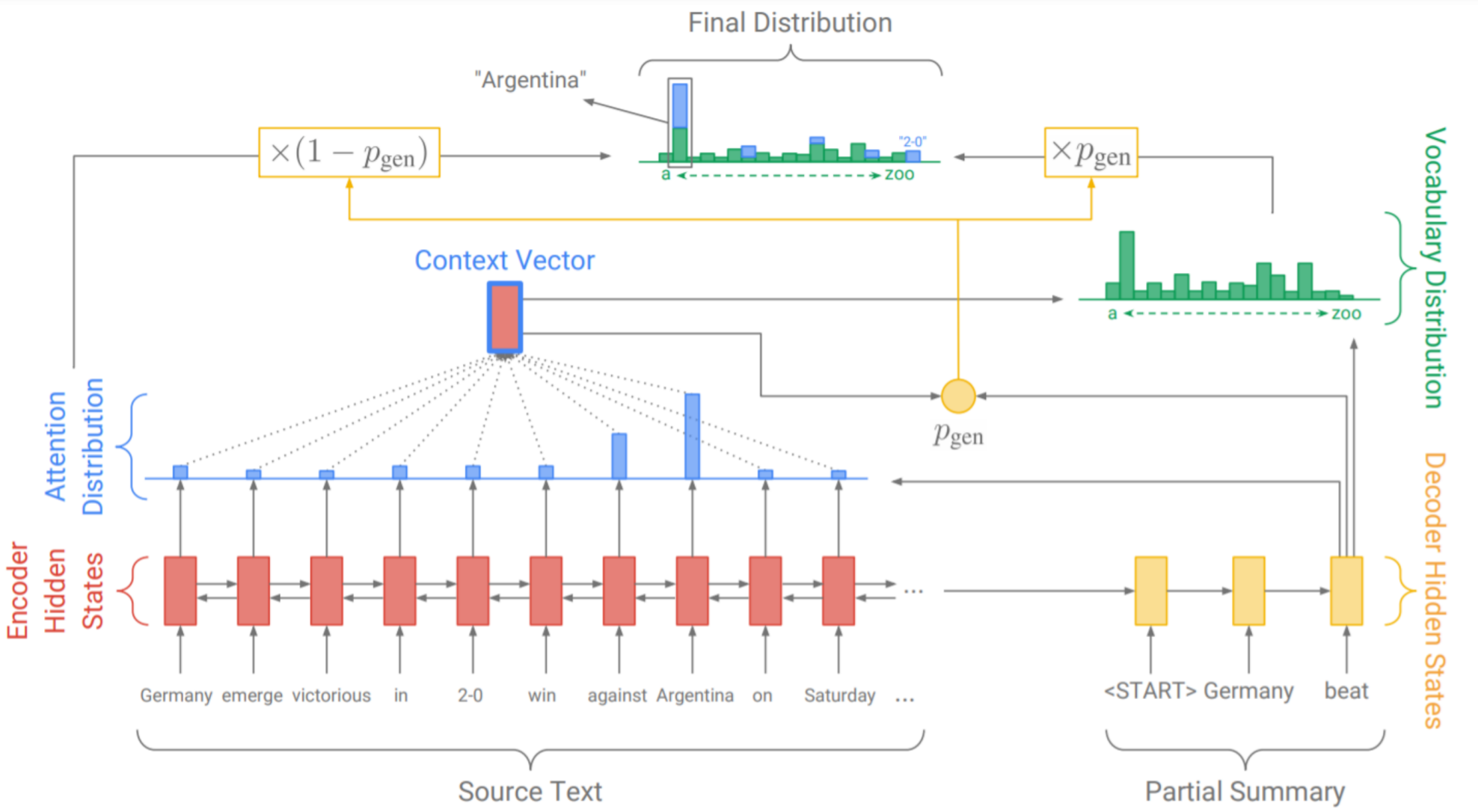
Пересчёт скрытых состояний декодировщика по $y = (y_1, \dots, y_m)$:

$$z_j = GRU_{dec}\left(Emb(y_i), [z_{j-1}, c_j]\right)$$

Выдача итоговых вероятностей:

$$p_{model}\left(y_j \middle| x, y_{<j}\right) = softmax_{y \in \mathrm{Y}}\left(W\left[c_j, z_j\right] + b\right)$$

Pointer-generator network



Pointer-generator network

Зададим вероятность копирования из исходного текста вместо самостоятельной генерации следующим образом:

$$p_{gen} = \sigma\left(\langle u, [c_j, z_j, Emb(y_j)]\rangle + \hat{b}\right)$$

Для задания распределения на копируемые токены будем использовать веса внимания:

$$p_{copy}(y_j|x, y_{<j}) = \sum_{i=1}^{n} \mathbb{I}[x_i = y_j]\alpha_{ij}$$

Итоговая вероятность задаётся как смесь двух распределений:

$$p(y_j|x, y_{<j}) = p_{gen}\ p_{model}(y_j|x, y_{<j}) + (1 - p_{gen})p_{copy}(y_j|x, y_{<j})$$

[See et al (2017); Get To The Point: Summarization with Pointer-Generator Networks]

Регуляризация механизма внимания

Модель явно подмешивает значения внимания в выходное распределение. Хотим явно запретить механизму внимания концентрироваться вокруг одних и тех же токенов.

Добавим в механизм внимания учёт прошлых шагов (на примере аддитивного механизма внимания):

$$\mathrm{A}_j^i = \sum_{k=0}^{j-1} \alpha_{ik}\,, \qquad A_j \in \mathbb{R}^{|x|}$$

$$sim\left(h_i, z_{j-1}\right) = v^t \tanh\left(W_h h_i + W_z z_{j-1} + \left\langle w_a, A_j \right\rangle + \tilde{b}\right)$$

Bottom-Up Attention

Проблема. На практике pointer-generator сети могут копировать слишком много информации. Одно из решений – явный запрет копирования некоторых токенов.

1. Решаем задачу разметки на отдельных токенах (входит / не входит в абстракт)
2. В распределении p_{copy} зануляем вероятности всех “не входящих” в абстракт токенов

Gehrmann et al (EMNLP 2018); Bottom-Up Abstractive Summarization

Суммаризация как задача языкового моделирования

Вместо энкодер-декодер архитектуры можно использовать языковое моделирование.

Предлагается для обучения суммаризации научных статей подавать на вход модели последовательность формата $x = (i_1, \dots, i_n, s_1 \dots, s_m, a_1, \dots a_k, d_1 \dots, d_N)$, где:

- i – предложения из introduction статьи
- s – предложения выделенные моделью extractive суммаризации
- a – предложения абстракта статьи
- d – оставшиеся предложения документа

S.Subramanian et al (2019); On Extractive and Abstractive Neural Document Summarization with Transformer Language Models

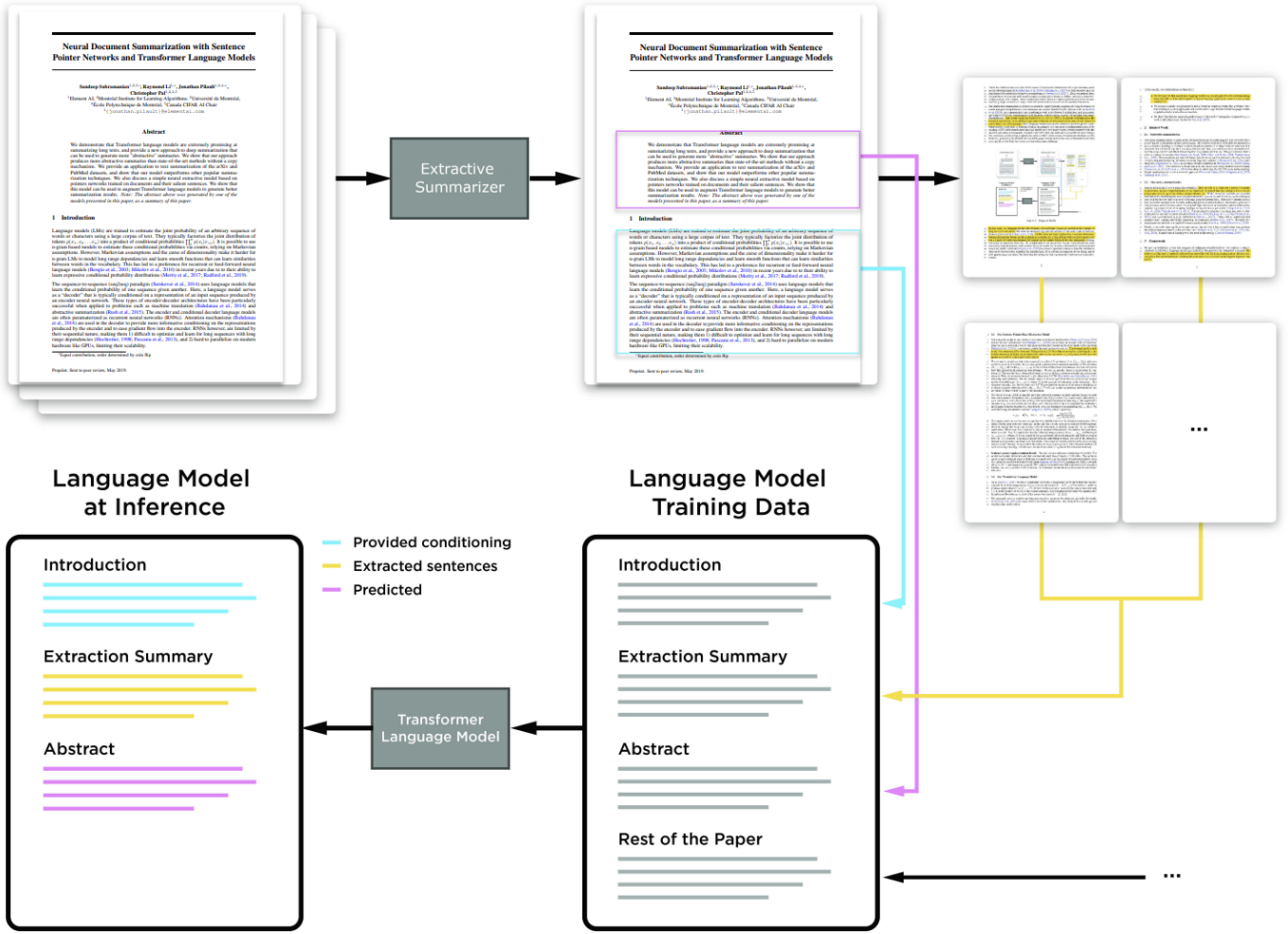
Почему такой формат последовательности работает?

$$x = (i_1, \dots, i_n, s_1 \dots, s_m, a_1, \dots a_k, d_1 \dots, d_N)$$

$$\sum_{x_j} \log p\left(x_j \middle| x_{<j}\right) \rightarrow \max$$

- учимся сжимать информацию: по introduction и extractive суммаризации восстанавливаем абстракт
- учимся распаковывать информацию: по introduction, extractive сумаризации и абстракту восстанавливать весь документ

Суммаризация как задача языкового моделирования



[S.Subramanian et al (2019); On Extractive and Abstractive Neural Document Summarization with Transformer Language Models]

Перенос обучения

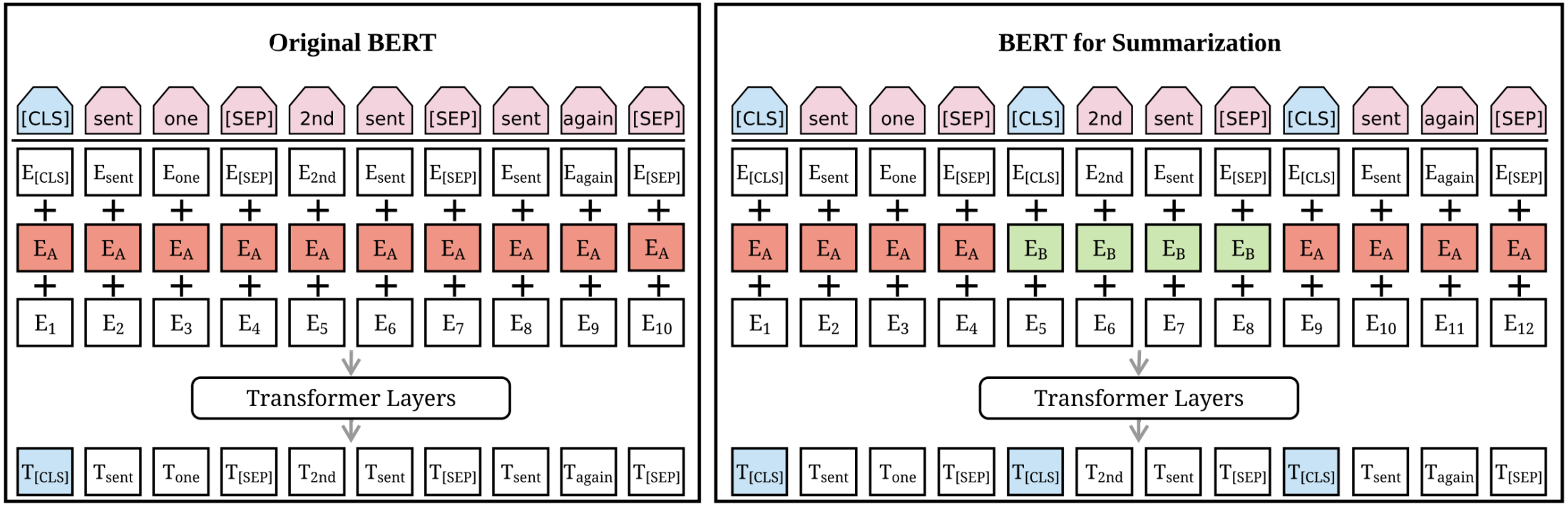
Один из способов бороться с недостатком данных – использовать перенос обучения:

1. Использование предобученного энкодера (BERT) с подходом преобразования последовательности
2. Использование предобученного энкодера-декодера (T5) с подходом преобразования последовательности
3. Использование предобученного декодера (GPT) с подходом языкового моделирования

BertExtAbsSum – адаптация BERT для суммаризации

В начале каждого предложения добавляем [CLS] токен.

Чередуем значения сегментных эмбеддингов для соседних предложений.



[Liu and Lapata (2019); Text Summarization with Pretrained Encoders]

Сильный энкодер vs слабый декодер

Так как энкодер – предобученный, а декодер инициализируется случайно, обучение может быть нестабильным.

Чтобы избежать этого можно задавать разные параметры оптимизатора для каждой части сети:

- у декодера выше темп обучения (например, 0.1 vs 2e-3)
- у декодера меньше шагов разогрева темпа обучения (например, 10к vs 20к)

Трёхэтапное обучение

Перед обучением всей архитектуры, мы можем попробовать адаптировать энкодер под задачу.

Будем решать энкодером задачу extractive суммаризации в стиле разметки, делая предсказания на [CLS] токенах предложений.

Model	R1	R2	RL
ORACLE	49.18	33.24	46.02
LEAD-3	39.58	20.11	35.78
Extractive			
COMPRESS (Durrett et al., 2016)	42.20	24.90	—
SUMO (Liu et al., 2019)	42.30	22.70	38.60
TransformerEXT	41.95	22.68	38.51
Abstractive			
PTGEN (See et al., 2017)	42.47	25.61	—
PTGEN + COV (See et al., 2017)	43.71	26.40	—
DRM (Paulus et al., 2018)	42.94	26.02	—
TransformerABS	35.75	17.23	31.41
BERT-based			
BERTSUMEXT	46.66	26.35	42.62
BERTSUMABS	48.92	30.84	45.41
BERTSUMEXTABS	49.02	31.02	45.55

Model	R1	R2	RL
ORACLE	29.79	8.81	22.66
LEAD	16.30	1.60	11.95
Abstractive			
PTGEN (See et al., 2017)	29.70	9.21	23.24
PTGEN+COV (See et al., 2017)	28.10	8.02	21.72
TCONVS2S (Narayan et al., 2018a)	31.89	11.54	25.75
TransformerABS	29.41	9.77	23.01
BERT-based			
BERTSUMABS	38.76	16.33	31.15
BERTSUMEXTABS	38.81	16.50	31.27

Энкодер-декодер для переноса обучения

Хотим обучить большую модель энкодер-декодер, чтобы использовать её для переноса обучения.

Проблема. Большинство рассмотренных нами моделей переноса обучения обучались на огромных выборках, составленных из неразмеченных данных.

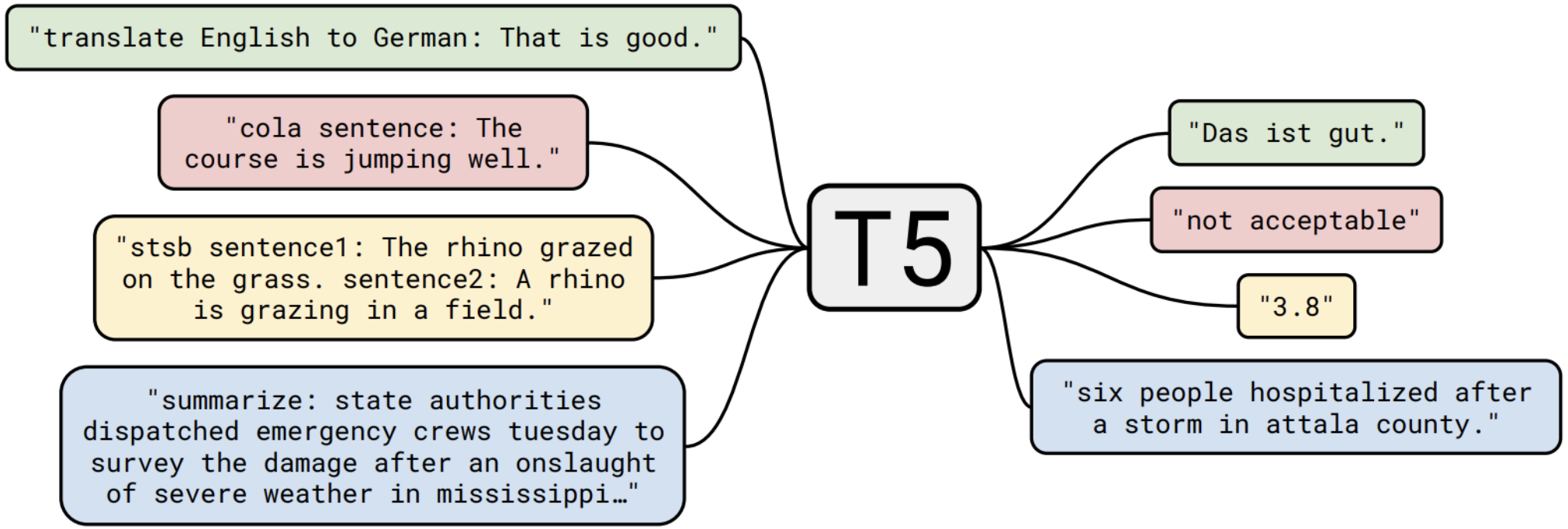
Где взять большое количество данных для обучения в формате преобразования последовательности?

Multitask подход

1. Будем формулировать все задачи в стиле преобразования последовательности (text-to-text подход).
2. Будем строить единую модель решающую все задачи одновременно (multitask подход).

Задача	Вход	Выход
Машинный перевод (датасет WMT)	“translate English to German: That is good.”	“Das it gut”
Анализ ошибок (датасет COLA)	“cola sentnece: the course is jumping well”	“not acceptable”
Суммаризация (датасет daily mail)	“summarize:marouane fellaini and adnan januzaj continue to…”	“the belgian duo took…”

Модель T5

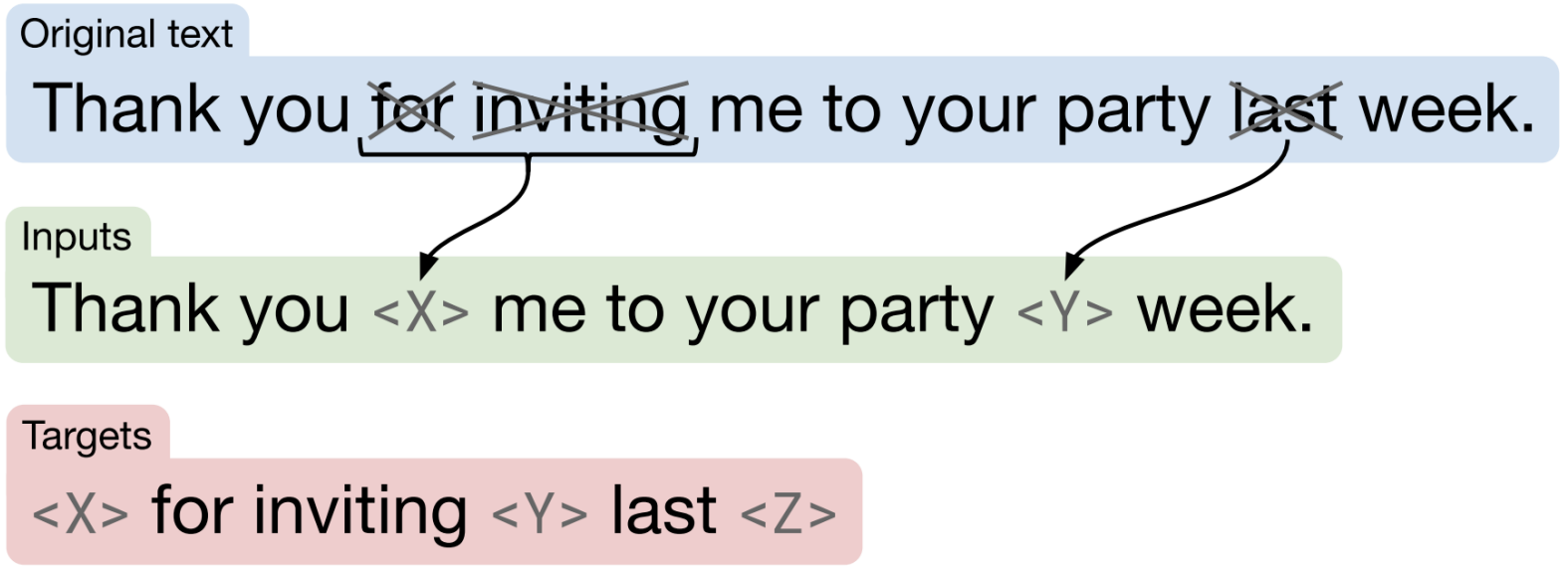


[Raffel et al (2020); Exploring the Limits of Transfer Learning with a Unified Text-to-Text Transformer]

Задачи предобучения без учителя

Перед обучением с учителем предлагается использовать дополнительное переобучение без учителя.

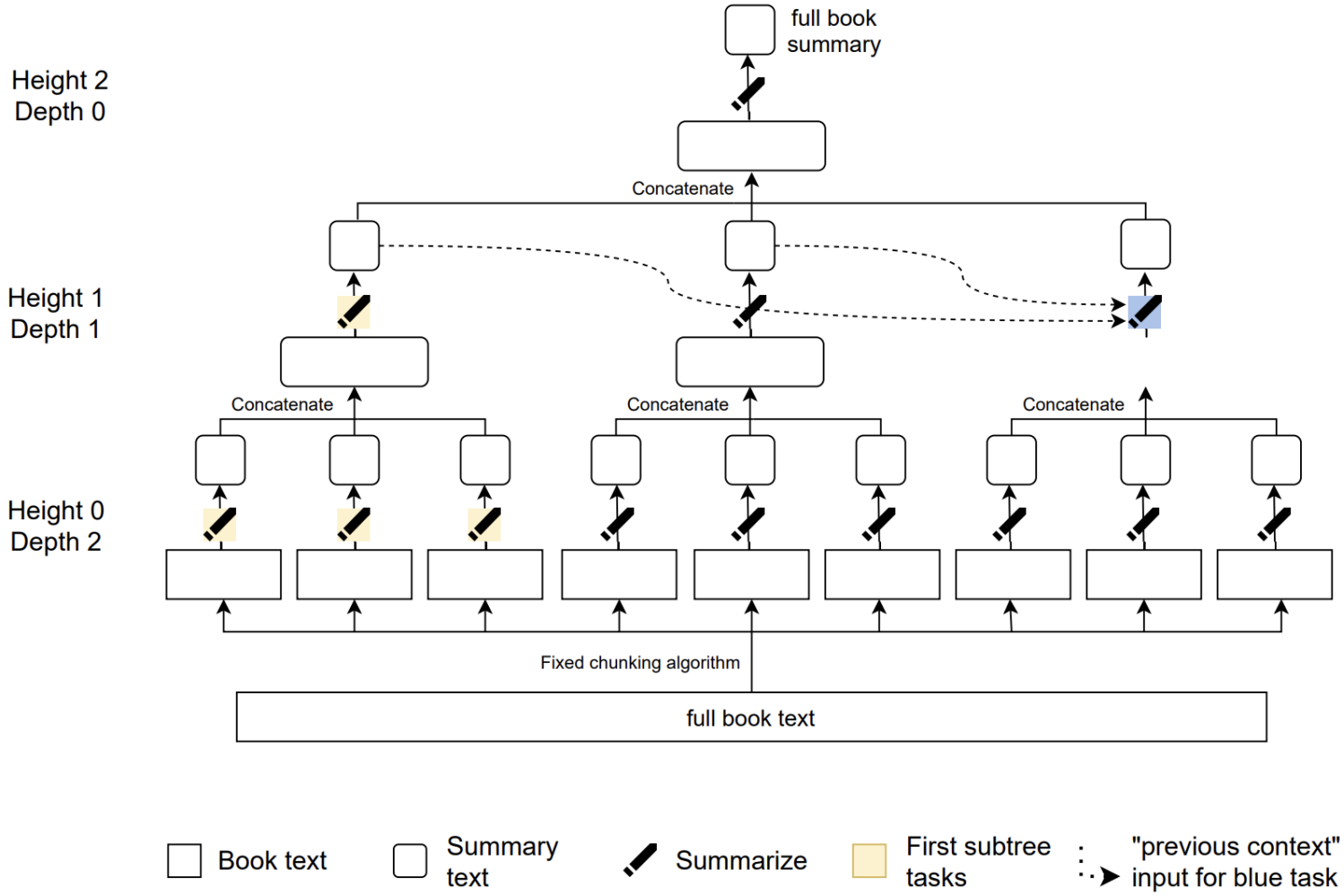
1. Заменяем некоторые подпоследовательности на специальные токены (уникальный для каждой подпоследовательности)
2. Восстанавливаем пропущенные подпоследовательности



T5: некоторые детали архитектуры

- Почти полностью соответствует архитектуре классического трансформера
- Энкодер и декодер состоят из 12 блоков каждый
- Размер внутреннего слоя 768
- Дополнительно используют дропаут “везде где он может быть применён”
- Во всех блоках используются относительные позиционные эмбеддинги с одним набором параметров
- Предообучается больше чем на 750GB данных
- Весами предобученной модели можно инициализировать энкодер-декодер и дообучать его под любую задачу

Суммаризация больших фрагментов текста



[Wu et al (2021); Recursively Summarizing Books with Human Feedback]

Резюме по abstractive суммаризации

- Всё ещё открытая задача, но с каждым годом решается всё лучше и лучше
- Хорошо работают подходы, комбинирующие extractive и abstractive подходы
- Перенос обучения может быть очень полезен в силу отсутствия обучающих выборок большого размера