Reflections of syntactic structures in non-autoregressive language models

Sergey Pletenev
HSE
Moscow
alex010rey@gmail.com

Abstract

Since the popularization of the Transformer as a general-purpose feature encoder for NLP, many studies have attempted to decode linguistic structure from its novel multi-head attention mechanism. However, much of such work focused almost exclusively on autoregressive style of decoding. In this study, we present decoding experiments on Non-autoregessive models in order to test the generalizability of the claim that dependency syntax is reflected in attention patterns.

Keywords: Machine translation, attention, seq2seq, nlp
DOI: 10.28995/2075-7182-2021-20-1180-1187

Отражения синтаксических структур в неавторегрессионных языковых моделях

Сергей Плетенев
НИУ ВШЭ
Москва
alex010rey@gmail.com

Аннотация

С момента появления архитектуры Трансформер в качестве универсального кодировщика языковых признаков, множество исследований рассматривали возможность декодирования лингвистических структур из данной архитектуры. Однако, большая часть исследований была сосредоточена только на авторегрессионном стиле декодирования. В данном исследовании мы представляем эксперименты по декодированию на неавторегрессионных моделях, чтобы проверить утверждение о том, что синтаксис зависимостей отражается в паттернах внимания.

Ключевые слова: Машинный перевод, механизм внимания, seq2seq, nlp

1 Введение

Методы обработки естественного языка (Natural Language Processing, NLP) и компьютерной лингвистики используются во многих сферах. Основные задачи, которые решаются с помощью подобных систем это: поиск по тексту, классификация документов, машинный перевод, генерация текста по заданым параметрам и многое другое.

В данный момент появляется множество работ, посвященных нахождению математических и лингвистических причин качества работы языковых моделей. Предлагаются идеи нахождения языковых зависимостей внутри механизма внимания. С помощью анализа механизма внимания пытаются определить, как обученная модель реагирует на различные синтаксические связности (анафора и кореференция) [3], редкие слова, структуры последовательности слов (Субъект-глагол-объект и другие) [13]

Однако подобный анализ делают в основном лишь для классических авторегрессионных моделей(auto-regressive-model, AR). Но кроме таких моделей существует и большой пласт неавторегрессионных моделей (Non-auto-regressive model, NAR) [9][11]. Главное различие между двумя

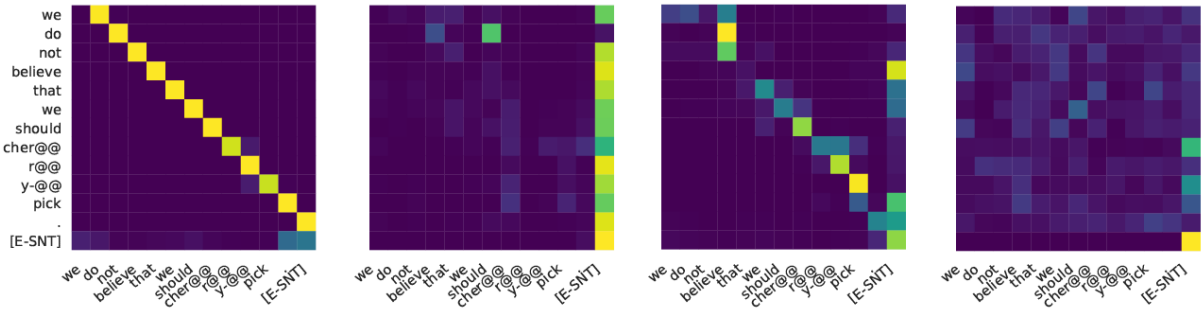

Рис. 1: Пример диагонального, вертикального, диагонального-вертикального и другого паттерна

моделями в том, что AR генерирует текст последовательно, слово за слово, а NAR генерирует все токены текста одновременно и параллельно. Неавторегрессионные модели чаще всего не достигают качества авторегрессионных моделей, но благодаря параллельному генерированию текста значительно обгоняют авторегрессионные модели по скорости, что важно при использовании в реальных индустриальных системах.

В данной работе производится анализ и исследование свойств NAR моделей. Основная часть работы состоит из изучения и сравнения существующих методов анализа авторегрессионных моделей машинного перевода. Эти методы будут переложены на неавторегрессионные модели. [1]

2 Методология

Как уже было сказано в предыдущей главе, неавторегрессионные модели хорошо ускоряют перевод текстов. Взамен, качество и точность перевода падает, хотя архитектура остается прежней. Можно предположить, что это происходит из-за отсутствия у моделей некоторых внутренних структур, таких как внимание на уровне декодировщика.

В данной работе мы опираемся на методологию, которая была выработана в статьях [2][10]. В них описывается структура поиска языковых функций в кодировщиках авторегрессионных языковых моделей. В данной методологии сначала анализируются головы кодировщиков на значимость в модели. Это делают с помощью механизма прунинга, при котором последовательно из модели удаляются наиболее плохие (в зависимости от функции потерь) головы. Это позволяет получить степень важности каждой из голов для модели. Далее, каждая из голов анализируется на наличие интерпритируемых функций. Первая функция – позиционная, т.е. голова отвечает за выучивание последовательности токенов в тексте. Вторая – синтаксическая, и отвечает за связи токенов между собой. Мы перенесли данные исследования на структуру NAR.

В первую очередь рассмотрим матрицы внимания в разных моделях и сравним часто встречаемые в них паттерны.[12] Цель данного эксперимента – проверить схожи ли паттерны, которые выучивает модель, существует ли уникальные паттерны для каждой из архитектур, и найти зависимость между паттернами и качеством перевода.

В статье [12] были выделены следующие паттерны у матриц внимания (Пример на рисунке 2):

• Диагональный паттерн – наибольший вес расположен на главной диагонали матрицы, или на сдвиге +1/-1 от главной диагонали

• Вертикальный паттерн – наибольший вес расположен на одной из колонок матрицы

• Диагонально-вертикальный паттерн – объеденение диагонального и вертикального паттернов

• Другой паттерн – все те матрицы, которые не входят в первые три паттерна

Также модели тестируются на нескольких задачах: кореференция и синтаксические структуры. Кореференция – это обозначение одного и того же объекта в разных частях текста. Речь идет о

[1] Код экспериментов доступен на: https://github.com/A1exRey/ReflectionOfNAR

следующих явлениях: повторы одного и того же полнозначного наименования, использование синонимов, антонимов и так далее.

Для тестирования решения задачи кореференции используется предобученая модель [7], с помощью которой из тестового набора WMT-16 En-De выделено 1000 примеров с кореференецией. Проверка на наличие кореференции происходит с помощью индексирования двух референтов в матрице внимания. Пусть индексы референтов в тексте будут i, j, а матрица внимания h будет размерностью $[src \times src]$, где src размер исходного текста. Тогда, если верно условие $h[i, j] + h[j, i] \geq 0.5$ то матрица внимания нашла кореференцию.

Для тестирования синтаксических структур были также выделены из тестового набора все тройки объектов субъект-глагол-объект (subject-verb-object) с помощью заранее обученой модели от AllenNLP [1], которая обучена на задаче OpenIE. Данная модель выделяет из текста Метод поиска синтаксических структур полностью повторяет поиск кореференции. Пусть индексы референтов в тексте будут $i_{obj}, i_{verb}, i_{subj}$, а матрица внимания h будет размерностью $[src \times src]$, где src размер исходного текста. Тогда, если верно хотя бы одно из условий $h[i_{obj}, i_{verb}] \geq 0.5$, $h[i_{obj}, i_{subj}] \geq 0.5$, $h[i_{verb}, i_{subj}] \geq 0.5$ то матрица внимания правильно выделила структуру.

3 Модели

Для исследования были выбраны три архитекутры: Transformer , Conditional Masked Language Model (CMLM) и Disentangled Context Transformer (DisCo).

Все наши модели обучаются с использованием метода дистилляции знаний [8]. Моделью-учителем выступает авторегрессионный Transformer En-De [4]. При тесте для всех моделей используются параметр beam-search = 5, для модели CMLM используется алгоритм mask-predict с количеством итераций 10, для модели DisCo выбран алгоритм easy-first с количеством итераций 10.

3.1 AR

Для начала опишем работу обычной, авторегрессионной модели на примере архитектуры Transformer. Пусть X и Y будет набором всех исходных и переведенных текстов. В стандартной модели машинного перевода, все наборы тренировочных данных представляют собой элементы вида $(x, y) \in X \times Y$, то есть паралельные тексты. Основной задачей модели становится нахождение наилучшего распределения $p(y|x)$. Авторегрессионые модели решают подобную задачу с помощью разбиения распределения на множество последовательностей слева направо x и y_t :

$$p(y|x) = \prod_t p(y_t|x, y_{<t}) \tag{1}$$

Подобные структуры также используются в задачах языкового моделирования, где основная цель это получение наиболее качественного распределения. Пересчет слева направо удобен тем, что позволяет точно вычислять логарифмическую функцию правдоподобия. Благодаря этому возможно использовать различные апроксимирующие алгоритмы, такие как beam-search [6] и жадное декодирование:

$$\hat{y}_t = argmax\, p(y|x, \hat{y}_{<t}) \tag{2}$$

Но у авторегрессионного подхода есть минусы. Во-первых, для генерации требуется доступ ко всем элементам последовательности x, они не могут прерываться. Во-вторых, мы не можем использовать подобную архитектуру для задачи заполнения пропущенных элементов последовательности. И в-третьих, стандартная реализация подобного алгоритма требует n шагов генерации для получения n элемонтов, что значительно замедляет перевод больших объемов текста.

Сама модель Transformer состит из кодировщика-декодировщика с механизмом внимания. Механизм внимания избирательно фокусируется на частях исходного предложения во время перевода, а веса внимания определяют силу, с которой эти части влияют на пропорции, с которыми объединяется информация из разных позиций. Механизм внимания сначала вычисляет веса внимания: т.е. для каждого слова он вычисляет распределение по всем словам (включая себя). Затем

src	we do not believe that we should cher_ r_ y-_ pick .										
1	[mask]	[mask]	nicht	[mask]	[mask]	[mask]	[mask]	[mask]	[mask]	sollten	.
2	[mask]	[mask]	nicht	,	[mask]	wir	[mask]	[mask]	[mask]	sollten	.
3	[mask]	glauben	nicht	,	dass	wir	[mask]	[mask]	[mask]	sollten	.
4	wir	glauben	nicht	,	dass	wir	uns	[mask]	[mask]	sollten	.
5	wir	glauben	nicht	,	dass	wir	uns	aus_	suchen	sollten	.

Таблица 1: Пример генерации модели CMLM с алгоритмом Mask-predict. Токены <pad> и [EOS] убраны для наглядности

src	cher_ r_ y-_ pic_ king is a practice of taking only the most beneficial items .															
1	Kir_	r_	y-_	pic_	pic_	ist	ist	Praxis	Praxis	Praxis	nur	nütz_	nütz_	nütz_	Gegenstände	Gegenstände	nehmen	.
2	Kir_	sch_	y-_	pic_	king	ist	eine	Praxis	,	,	,	die	lichsten	lichsten	lichsten	Gegenstände	aufzunehmen	.
3	Kir_	sch_	ern_	pic_	king	ist	eine	Praxis	,	nur	nur	nur	nütz_	nütz_	esten	Gegenstände	aufzunehmen	.
4	Kir_	sch_	ern_	pic_	king	ist	eine	Praxis	,	nur	die	die	die	haft_	esten	Gegenstände	aufzunehmen	.
5	Kir_	sch_	ern_	pic_	king	ist	eine	Praxis	,	nur	die	vorteil_	vorteil_	nütz_	esten	Gegenstände	aufzunehmen	.
6	Kir_	sch_	ern_	pic_	king	ist	eine	Praxis	,	nur	die	vorteil_	haft_	haft_	esten	Gegenstände	aufzunehmen	.
7	Kir_	sch_	ern_	pic_	king	ist	eine	Praxis	,	nur	die	vorteil_	haft_	wert_	esten	Gegenstände	aufzunehmen	.

Таблица 2: Пример генерации модели DisCo с алгоритмом easy-first. T=10, но модель остановилась после 7 итерации. Токены <pad> и [EOS] убраны для наглядности

это распределение используется для вычисления нового представления этого слова, которое далее используется в последующих слоях. При многоголовом механизме внимании этот процесс повторяется несколько раз с разными репрезентациями, а результат объединяется.

3.2 NAR

3.2.1 Conditional Masked Language Model

Первая модель – Mask-Predict: Parallel Decoding of Conditional Masked Language Models (CMLM)[9]. Модель CMLM использует стандартную архитектуру Transformer с небольшим изменением: из-за неавторегрессионной структуры модели у декодировщика была убрана маска, которая предотвращает возможность модели смотреть на будущие токены. Кроме того, в модели используется специальный токен [LENGTH], так как NAR требуется знать окончательную длину предложения еще до старта декодирования.

Само декодирования происходит с помощью специального алгоритма mask-predict. На каждой итерации алгоритм маскирует некоторое подмножество токенов, и затем пытается их предсказать (параллельно) с помщью архитектуры CMLM. Процедура маскировки позволяет модели возвращаться и заново маскировать наиболее сложные случаи, но основываясь на хорошо предсказаных незамаскированых результатах перевода.

3.2.2 Disentangled Context Transformer

Вторая модель – Disentangled Context Transformer (DisCo) [11]. Данная модель является идейным продолжением предыдущей архитектуры. Но в отличии от CMLM, где модель может предсказывать только замаскированные слова, DisCo может предсказывать все токены одновременно, что дает ускорение перевода и улучшение качества модели. Также используется новый алгоритм easy-first, в котором каждое слово предсказывается по тем словам, в которых модель наиболее уверена. Этот алгоритм декодирования позволяет предсказывать в каждой итерации разные контексты для всех доступных токенов, что позволяет останавливать декодирование, когда модель получила хорошее предсказание. Этим данный алгоритм отличается от mask-predict, в котором количество итераций всегда задано некоторой константой T.

Алгоритм easy-first работет следующим образом. На первой итерации $(t = 0)$ параллельно предсказываются все токены в исходном тексте:

$$Y_n^1, p_n = (arg)max\ P(Y_n = w|X)$$

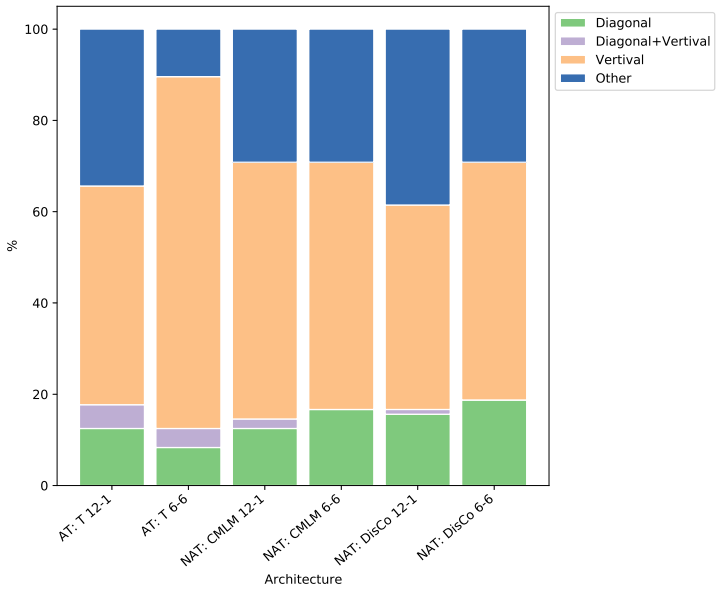

Рис. 2: Распределение паттернов кодировщиков по моделям

Далее алгоритм ранжирует полученую матрицу p_n от большего к меньшему и получает отсортированый список z, где $z(i) = rank\ p_i$. На последующих итерациях $(t > 1)$ происходит предсказание токенов с помощью:

$$Y_{obs}^{n,t} = \left\{ Y_t^{t-1} | z(i) < z(n) \right\}$$

$$Y_n^t, p_n^t = (arg)max\ P\left(Y_n = w | X, Y_{obs}^{n,t} \right)$$

Видно, что при каждой итерации генерируются токены на основании наиболее простых предсказаний. В данной статье описывается еще несколько алгоритмов декодировки, но easy-first оказался наиболее эффективным.

4 Эксперименты

В качестве набора данных для экспериментов используется параллельные корпус текстов для машинного перевода WMT14 EN-DE (4.5M пар предложений)[5].

4.1 Паттерны

Рассчитаем для всех наших моделей количество различных паттернов. Для этого напишем простую функцию, которая будет выбирать матрицы внимания, если они похожи с заданным паттерном хотя бы на половину. Результат работы функции виден на рисунке 4.1. Во всех моделях, кроме стандартной модели Transformer 6E-6D наблюдается схожее распредление различных паттернов.

4.2 Кореференция

Мы рассмотрели только синтаксические структуры нашей модели: индексирование и связь текста при переводе. Кроме формальных структур в архитектуре хорошей модели перевода должны находиться и иные признаки. Стандартные системы машинного перевода обрабатывают предложения изолированно и, следовательно, игнорируют лишнюю информацию, даже если расширенный контекст может как предотвратить ошибки в двусмысленных случаях, так и улучшить согласованность перевода.

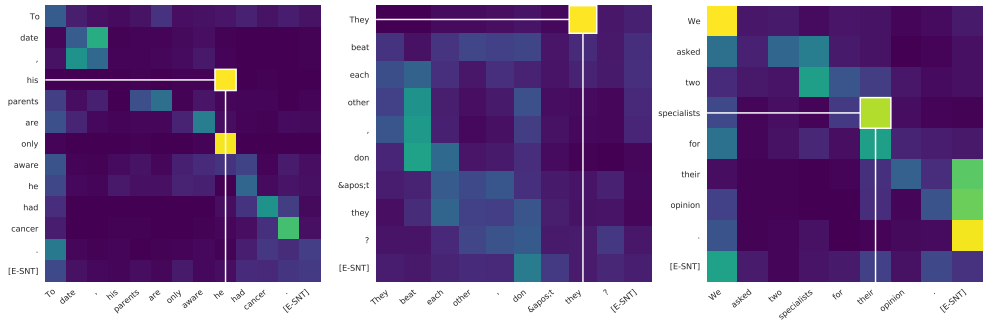

Рис. 3: Пример нахождения кореференции моделью CMLM 6-6

Предыдущий эксперимент мог не показать головы, которые отвечают за решение кореференции в качестве главной или важной. Поэтому составим на основе наших тестовых данных новый набор предложений с гарантированной кореференцией и проанализируем головы. В главе 2 описан полностью алгоритм, с помощью которого мы будем анализировать матрицы внимания на наличие кореференции. Пример нахождения моделью кореференции показан на рисунке 4.2

4.3 Триплеты

Мы предполагаем, что при выполнении перевода, кодировщик архитектуры Transformer может быть ответственен за устранение неоднозначности синтаксической структуры исходного предложения. Поэтому мы хотим знать, обращают ли головы внимания на лексемы, соответствующие какому-либо из основных синтаксических отношений в предложении. В нашем анализе мы рассмотрим отношения субъект-глагол-объект (subject –verb –object, SVO) для наших текстов. Полная процедура генерации и создания проверочных текстов описана в главе 2.

5 Результаты

Размерность	E6-D6	E12-D1
Transformer	33.58	33.06
CMLM	31.83	27.19
DisCo	32.46	29.28

Таблица 3: Модели для экспериментов, точность указана в BLEU

Все полученные модели показали уже известный факт: авторегрессионный подход превосходит по качеству BLEU любые NAR модели.[4] Эти данные подтверждаются на таблице 3. Тестировались 3 различные архитектуры с разным количеством слоев кодировщика-декодировщика: 6-на-6 и 12-на-1.

Проанализируем полученые результаты. Рассмотрим таблицу 4. В данной таблице указана точность решения задачи кореференции а также процент голов, в котором алгоритм нашел кореференцию. Мы разделили такие головы на три группы: в которых никогда не находилась кореференция(=0), в которых редко появлялась кореференция (<0.1) и часто (>0.1). Как мы видим, обычная авторегрессионная модель показала самые низкие результаты. У модели мало голов для решения кореференции. Наоборот же, неавторегрессионные модели показали лучший результат. Само решение задачи кореференции разложено на множество голов. Некоторые головы сработали всего несколько раз. Возможно это связано с тем, что неавторегрессионые модели в качестве обучения

Модель	Точность	Количество голов		
		=0	<0.1	>0.1
Transformer E12-D1	0.06	84	0	12
CMLM E12-D1	0.17	58	26	11
DisCo E12-D1	0.19	53	32	11
Transformer E6-D6	0.04	39	0	9
CMLM E6-D6	0.16	35	9	4
DisCo E6-D6	0.16	31	13	4

Таблица 4: Точность решения задачи кореференции

решают задачу восстановления текста, тем самым им требуется учить более сложные языковые структуры.

Модель	Точность	Количество голов		
		0	<0.1	>0.1
Transformer E12-D1	0.42	1	81	12
CMLM E12-D1	0.4	11	74	11
DisCo E12-D1	0.4	9	74	12
Transformer E6-D6	0.36	0	42	6
CMLM E6-D6	0.35	9	34	5
DisCo E6-D6	0.36	8	32	8

Таблица 5: Точность нахождения SVO

	Позиционная			Синтаксическая	
Модель	diag	vertical	other	Кореференция	O-V-S
Transformer 12-1	6	2	2	0	0
CMLM 12-1	8	1	1	4	4
DiSco 12-1	7	0	3	3	4

Таблица 6: Топ-10 наиболее важных голов

Результаты нахождение синтаксических структур показаны в таблице 5. В отличии от предыдушего эксперимента с кореференцией, в данном случае модели показали схожий результат по точности нахождения синтаксических структур внутри текста. Единственное существенное различие – количество голов, которые не влияют на поиск синтаксических структур. Их больше в неавторегрессионных моделях.

Кроме этого, с помощью механизма прунинга мы выделили топ-10 наиболее важных голов (Таблица 6). В неавторегрессионных моделях часть голов выполняют как синтаксические, так и позиционные функции одновременно. У авторегрессионных моделей синтаксические головы не входят в топ-10.

6 Заключение

Целью данной работы являлось нахождение закономерностей между моделями авторегрессионного и неавторегрессионного перевода. Мы провели комлпексный анализ трех различных моделей: Transformer, DisCo, CMLM. Исследование проводилось с помощью разносторонних экспериментов, которые анализировали различные особенности языка и перевода.

В частности, мы изучили внутренние паттерны кодировщиков неавторегрессионных моделей и доказали, что они по большей части схожи с паттернами из авторегрессионных моделей. Мы

также выяснили, что неавторрегрессионные модели лучше выделяют некоторые синтакстические особенности языка.

В результате мы опровергли изначальное предположение о том, что качество неавторегрессионных моделей страдает из-за того, что модель не находит схожие с авторегрессионными моделями различные языковые свойства.

References

[1] AllenNLP: A Deep Semantic Natural Language Processing Platform / Matt Gardner, Joel Grus, Mark Neumann et al. — 2017. — arXiv:1803.07640.

[2] Voita Elena, Talbot David, Moiseev Fedor et al. Analyzing Multi-Head Self-Attention: Specialized Heads Do the Heavy Lifting, the Rest Can Be Pruned. — 2019. — 1905.09418.

[3] Voita Elena, Serdyukov Pavel, Sennrich Rico, Titov Ivan. Context-Aware Neural Machine Translation Learns Anaphora Resolution. — 2018. — 1805.10163.

[4] Deep Encoder, Shallow Decoder: Reevaluating the Speed-Quality Tradeoff in Machine Translation / Jungo Kasai, Nikolaos Pappas, Hao Peng et al. // ArXiv. — 2020. — Vol. abs/2006.10369.

[5] Findings of the 2014 Workshop on Statistical Machine Translation / Ondrej Bojar, Christian Buck, Christian Federmann et al. // Proceedings of the Ninth Workshop on Statistical Machine Translation. — Baltimore, Maryland, USA : Association for Computational Linguistics, 2014. — June. — P. 12–58. — Access mode: http://www.aclweb.org/anthology/W/W14/W14-3302.

[6] Freitag Markus, Al-Onaizan Yaser. Beam Search Strategies for Neural Machine Translation // Proceedings of the First Workshop on Neural Machine Translation. — 2017. — Access mode: http://dx.doi.org/10.18653/v1/W17-3207.

[7] HuggingFace's Transformers: State-of-the-art Natural Language Processing / Thomas Wolf, Lysandre Debut, Victor Sanh et al. // ArXiv. — 2019. — Vol. abs/1910.03771.

[8] Kim Yoon, Rush Alexander M. Sequence-Level Knowledge Distillation // Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing. — Austin, Texas : Association for Computational Linguistics, 2016. — Nov. — P. 1317–1327. — Access mode: https://www.aclweb.org/anthology/D16-1139.

[9] Marjan Ghazvininejad Omer Levy Yinhan Liu Luke Zettlemoyer. Mask-Predict: Parallel Decoding of Conditional Masked Language Models // Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing. — 2019.

[10] Michel Paul, Levy Omer, Neubig Graham. Are Sixteen Heads Really Better than One? — 2019. — 1905.10650.

[11] Kasai Jungo, Cross James, Ghazvininejad Marjan, Gu Jiatao. Non-Autoregressive Machine Translation with Disentangled Context Transformer. — 2020. — 2001.05136.

[12] Revealing the Dark Secrets of BERT / Olga Kovaleva, Alexey Romanov, Anna Rogers, Anna Rumshisky. — 2019. — 1908.08593.

[13] Voita Elena, Sennrich Rico, Titov Ivan. When a Good Translation is Wrong in Context: Context-Aware Machine Translation Improves on Deixis, Ellipsis, and Lexical Cohesion. — 2019. — 1905.05979.