Исследование возможности одновременного обучения диффузионной модели и текстового автокодировщика

A Preprint

Ким Роман Германович[1, 2]
Научный консультант: Мещанинов Вячеслав Павлович[1]
Научный руководитель: Темирчев Павел Георгиевич[2]

[1]Национальный исследовательский университет «Высшая школа экономики»
[2]Московский государственный университет имени М. В. Ломоносова
vmeshchaninov@hse.ru, rkim@hse.ru

19 декабря 2024 г.

Abstract

Генеративные модели занимают важное место в современном машинном обучении и обработке сигналов, применяясь для синтеза изображений, аудио, 3D-объектов и текста. С течением времени подходы к генерации данных стали более разнообразными и мощными: от Генеративно-состязательных сетей (GAN) (1) до Вариационных Автоэнкодеров (VAE) (2; 3), поточных моделей (4; 5) и энергетических моделей (6), а в последние годы – диффузионных моделей (7; 8). В частности, диффузионные модели продемонстрировали превосходство над классическими методами генерации изображений, а впоследствии стали активно развиваться в задачах генерации текстов (9; 10; 11; 12; 13; 14). Однако переход к дискретным пространствам, характерным для текстовых данных, ставит новые вызовы, которые стимулируют поиск интегральных решений, совмещающих автокодировщики и диффузионные модели для упрощения задачи восстановления структуры и смысла текста.

Данная статья нацелена исследовать возможность одновременного обучения диффузионной модели и автокодировщика. Также будет предложены две конфигурации для одновременного обучения.

1 Введение

Современные генеративные модели охватывают широкий спектр методов:

- GAN (Generative Adversarial Networks): Предложенные в (1), GAN получили широкую популярность благодаря способности генерировать реалистичные изображения. Последующие работы улучшили стабильность обучения (Wasserstein GAN (15; 16)), качество (StyleGAN (17; 18)) и масштабируемость (19). Тем не менее, GAN страдают от проблем смещения, катастрофического забывания и mode collapse (20; 21).
- VAE (Variational Autoencoders): Модели на основе вариационных автоэнкодеров (2; 3) вводят вероятностный подход к обучению латентного представления. Однако качество сгенерированных ими изображений часто уступает GAN.
- Поточные модели (Flow-based models): Обращаемые преобразования плотности (4; 5) позволяют напрямую моделировать распределение данных, но требуют больших вычислительных ресурсов.
- Энергетические модели (EBM): Подходы (6) предлагают вероятностную интерпретацию через энергетический ландшафт, но обучение EBM нетривиально.

В последние годы особое внимание уделяется диффузионным моделям (8; 7; 22; 23; 24; 25; 26; 27; 28), продемонстрировавшим превосходство в широком спектре задач: от сверхразрешения изображений (29) и синтеза аудио (30; 31) до генерации 3D-объектов (32; 33), но при работе с текстами у диффузионных моделей происходят проблемы, так как они были придуманы для непрерывных данных (изображений, звука), а пространств слов - дискретное, вследствие чего приходится, либо переводить дискретное пространство текстов в непрерывное, либо адаптировать работу диффузионной модели на дискретные пространства. В данной статье будет рассматриваться непрерывная диффузионная модель, работающая с непрерывным пространством энкодингов.

2 Формальное описание задачи генерации текста

2.1 Классические методы генерации текста

Классическая задача генерации текста с помощью нейронных сетей формулируется в терминах максимизации правдоподобия текстов из обучающей выборки.

$$\prod_{x\in X} p_\theta(x) = \prod_{x\in X} p_\theta(x_1,\dots,x_{n_x}) = \prod_{x\in X}\left(p_\theta(x_1)\prod_{i=2}^{n_x} p_\theta(x_i \mid x_1,\dots,x_{i-1})\right) \to \max_\theta, \tag{1}$$

где x – текст из обучающей выборки X, а n_x – число слов в этом тексте. Для того, чтобы убрать безусловную вероятность $p_\theta(x_1)$ из функционала, в начало текста добавляют фиксированный символ начала последовательности $x_0 =$ “bos”. Тогда все вероятности становятся условными.

$$\prod_{x\in X}\left(\prod_{i=1}^{n_x} p_\theta(x_i \mid x_1,\dots,x_{i-1})\right) \to \max_\theta \tag{2}$$

Функционал ошибки, минимизируемый в процессе оптимизации получается из данного правдоподобия с помощью наложения на него логарифма и домножения на -1.

$$-\sum_{x\in X}\left(\sum_{i=1}^{n_x} \ln p_\theta(x_i \mid x_1,\dots,x_{i-1})\right) \to \min_\theta \tag{3}$$

В соответствии с полученным функционалом ошибки обучается нейронная сеть, которая принимает на вход последовательность слов и возвращает условное распределение на слова, которые могут быть сгенерированы в качестве продолжения. Процесс генерации текста в таких моделях является авторегрессионным. То есть слова генерируются по-очереди.

2.2 Диффузионные методы генерации текста

У авторегрессионной генерации есть три основных недостатка. Для того, чтобы сгенерировать текст длины n, необходимо вызвать модель n раз. Из-за этого генерация длинных текстов работает относительно долго.

Во-вторых, авторегрессионным моделям свойственна генерация повторяющихся фраз. Такое поведение возникает из-за оптимизируемого функционала. В ситуации, когда текст содержит повторы, вероятность появления дополнительно повтора возрастает и, вследствии этого, модель с каждым новым словом начинает чаще генерировать повторяющиеся фразы (34). Для исправления этого недостатка приходится подбирать параметры для выбора токенов из предсказанного распределения, что усложняет настройку модели.

И наконец, авторегрессионные модели не могут исправлять свои ошибки. Например, при генерации стихов из-за того, что модель не смотрит в будущее, она может сгенерировать слово, к которому нет подходящей рифмы. В этом случае весь следующий текст будет испорчен.

Диффузионные модели решают все эти проблемы. Они итеративно генерируют объекты из шума, приближая вероятность исходных данных. При применении к тексту такой метод позволяет генерировать все слова одновременно, так как модель воспринимает текст как единый объект. При этом для генерации требуется фиксированное число итераций (около 100), меньшее, чем необходимо для генерации длинного текста авторегрессионно.

Обучение диффузионной модели происходит с помощью стохастического градиентного спуска, в котором считается градиент функционала ошибки по параметрам модели. Так как текст имеет дискретную природу, если модель будет предсказывать его напрямую, то градиент посчитать будет невозможно. По этой причине, диффузионные модели обычно работают в пространстве непрерывных векторных представлений слов (35; 36). При обучении перед зашумлением каждое слово отображается в соответствующий вектор, а при генерации после последней итерации каждый сгенерированный вектор слова отображается обратно в текст.

2.3 Непрерывная диффузионная модель

Непрерывные диффузионные модели повторяют идею диффузионных моделей, применяемых для изображений и аудио. В них объект зашумляется с помощью вливания в него гауссовского шума.

$$q(x_t|x_s) = \mathcal{N}(x_t; \alpha_{t|s}x_s, \sigma^2_{t|s}I) \tag{4}$$

Для текста в числовом формате чаще всего используются векторные представления токенов фиксированной длины (10; 11; 37; 38; 13; 14; 12). Однако иногда модель обучают на симплексе (39; 40), представляя каждый токен в виде вектора $+k, -k^{|V|}$, где $+k$ стоит на месте индекса токена, а на всех остальных позициях стоит $-k$.

3 Постановка задачи и варианты решения

В данной статье рассматривается модель состоящая из автокодировщика и диффузионной модели. В настоящее время в большинстве работ по диффузионным моделям обучение автокодировщика и диффузионной модели происходит раздельно, но одновременное обучение модели упрощает общий конвейер обучение и потенциально снижает количество источников ошибок во время обучения.

Предлагаемые варианты решения:

3.1 Конфигурация № 1

$$x_0 = \text{Enc}(w) \tag{5}$$
$$x_0 = \text{Normalize}(x_0) \tag{6}$$
$$t \sim U[0,1], \varepsilon \sim \mathcal{N}(0, I) \tag{7}$$
$$x_t = \alpha_t x_0 + \sigma_t \varepsilon, \tag{8}$$
$$\hat{x}_0 = \text{Dif}(x_t, 0, t)] \tag{9}$$
$$\text{if } \zeta < \frac{1}{2} \text{ then} \tag{10}$$
$$\hat{x}_0 = model(x_t, SG[\hat{x}_0], t) \tag{11}$$
$$\hat{w} = Dec(\hat{x}_0) \tag{12}$$
$$\mathcal{L}_{dif} = ||x_0 - \hat{x}_0||^2 \to \min_{dif} \tag{13}$$
$$\mathcal{L}_{ae} = \text{CE}(w, \hat{w}) \to \min_{enc,dec} \tag{14}$$

В этой конфигурации используется две функции ошибки: L_{ae} – функция ошибки кодировщика и декодировщика, L_{dif} – функция ошибки диффузионной модели. Важно заметить, что L_{ae} не оптимизируется по параметрам диффузии, так как обратное приводит к коллапсу диффузионной модели. Также важно отметить, что автокодировщик и диффузионная модель связаны, так как вход декодировщика – это выход диффузионной модели.

3.2 Конфигурация № 2

$$x_0 = \text{Enc}(w) \tag{15}$$
$$x_0 = \text{Normalize}(x_0) \tag{16}$$
$$\tag{17}$$
$$t \sim U[0,1], \varepsilon \sim \mathcal{N}(0, I) \tag{18}$$
$$x_t = \alpha_t x_0 + \sigma_t \varepsilon, \tag{19}$$
$$\tag{20}$$
$$\hat{x}_0 = \text{Dif}(x_t, 0, t)] \tag{21}$$
$$\text{if } \zeta < \frac{1}{2} \text{ then} \tag{22}$$
$$\hat{x}_0 = model(x_t, SG[\hat{x}_0], t) \tag{23}$$
$$\tag{24}$$
$$\hat{w} = Dec(\hat{x}_0) \tag{25}$$
$$\tag{26}$$
$$\mathcal{L}_{dif} = ||x_0 - \hat{x}_0||^2 \to \min_{dif} \tag{27}$$
$$\mathcal{L}_e = \text{CE}(w, \hat{w}) + ||x_0 - SG[\hat{x}_0]||^2 \to \min_{enc,dec} \tag{28}$$

Вторая конфигурация похожа на первую, но в функции потерь автокодировщика добавляется слагаемое для большей связи диффузионной модели и автокодировщика. Важно отметить, что стохастический градиент убирать из L_e нельзя, так как кодировщик будет стремиться сделать все токены константными или уменьшить их норму до нуля.

4 Эксперименты

Для экспериментов использовались датасеты (Wikipedia (41), Rocstories, XSUM), различные метрики качества (PPL, MAUVE, Rouge, BERTScore, Meteor). За архитектуру диффузионной модели взята архитектура LLaMa (42): 12 голов, 12 слоев. Размер рассматриваемой модели $\sim$ 100 млн параметров

В секциях !!! эксперименты будут проводится с моделью в конфигурации 1, для задачи безусловной генерации, длина генерации 8 токенов

4.1 Влияние размера обучающих данных на качество модели

Обучим модель в первой конфигурации на задаче безусловной генерации текста на Rocstories (размер набора данных $\sim$ 119000) и на Wikipedia (размер набора данных $\sim$ 2млн.) Результаты представлины на Рис. 1

По Рис. 1 можно сделать вывод, что размера Rocstories недостаточно для корректного обучения модели, поэтому последующие эксперименты будут проводиться на наборе данных Wikipedia.

4.2 Важность и подбор нормализации

Как можно заметить в каждой конфигурации участвует нормализаци данных, в данной секции будет приведено теоретическое доказательство важности нормализации и подбор лучшего вида нормализации.

4.2.1 Обоснование

Рассмотрим модель без нормализации.

Получаем векторные представления слов $x_0 = \text{Enc}(w)$, пусть $||x_0|| \to \infty$, тогда $x_t = \alpha_t x_0 + \sigma_t \varepsilon \approx \alpha_t x_0$, $\hat{x}_0 = \text{Dif}(\alpha_t x_0, t) \approx x_0$, потому что диффузионная модель учится предсказывать x_0, следовательно $\hat{w} = Dec(x_0)$.

Из этого следует, что кодировщику выгодно увеличивать норму x_0, чтобы делать задачу декодировщика проще.

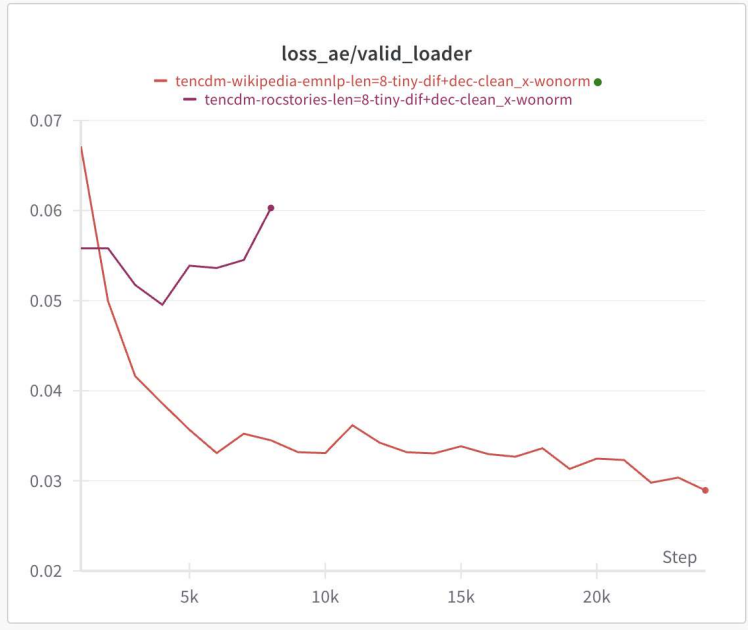

Рис. 1: Функция потерь диффузионной модели на валидационной выборке при одновременном обучении. (фиолетовая линия – Rocsories, оранжевая – Wikipedia)

4.2.2 Подбор типа нормализации

Для начала введем рассматриваемые типы нормализации:

1. Нормализация на сферу (SphereNorm)
2. Нормализация по куску данных (BatchNorm)
3. Нормализация по слою (LayerNorm)
4. Нормализация по всему набору данных (DataNorm)

Первые три варианта нормализации достаточно распространены и не требуют дополнительного объяснения, в свою очередь DataNorm может быть реализовано по-разному, а точнее может быть разная реализация сглаживания среднего отклонения, поэтому ниже будет представлен вариант сглаживания.

Рассмотрим, как можно сглаживать среднее отклонение:

$$x_n = \frac{1}{BS}\sum_{B,S} e_i \tag{29}$$

$$y_n = \frac{1}{BS}\sum_{B,S} e_i^2 \tag{30}$$

$$\tag{31}$$

$$\mu_n = (1-\alpha)\mu_{n-1} + \alpha x_n = \mu_{n-1} + \alpha(x_n - \mu_{n-1}) \tag{32}$$

$$q_n = q_{n-1} + \alpha * (y_n - q_{n-1}) \tag{33}$$

$$\tag{34}$$

$$\sigma_n = \sqrt{q_n - \mu_n^2} \tag{35}$$

Обучим модель с разными коэффициентами сглаживания для нормализации.

Из таблицы 1, можно сделать вывод, что модель с коэффициентом сглаживания 0.01 лучше по метрикам Mauve, Div, Memorization, при этом по метрике перплексия разница незначительная (0.4), в следствии чего, модель с коэффициентом сглаживания 0.01 выбрана лучшей.

Теперь сравним виды нормализации: По результатам из таблицы 2 BatchNorm показал себя хуже, чем

Таблица 1: Результаты модели обученной с DataNorm

α	PPL ↓	Mauve ↑	Div ↑	Memorization ↓
0.99	293.9	0.921	0.645	0.362
0.01	294.3	0.943	0.666	0.354

Таблица 2: Результаты модели с разными нормалиациями

Тип нормализации	PPL ↓	Mauve ↑	Div ↑	Mem ↓
Sphere Norm	319.5	0.932	0.672	0.324
Layer Norm	326.0	0.904	0.674	0.322
Data Norm	294.4	0.943	0.666	0.354
Batch Norm	344.1	0.934	0.705	0.319

остальные варианты нормализации. При этом DataNorm лучший вариант нормализации, поэтому в будущем будем использовать именно его.

4.3 Сравнение конфигураций с существующими диффузионными моделями

В данной секции будет произведено сравнение качества реализованных конфигураций и существующих диффузионных моделей на задачах суммаризации и безусловной генерации. Для сравнения взяты модели такого же размера, а именно LD4LG (43) и TEncDM (44)

4.3.1 Безусловная задача

Сравнение двух подходов с существующими моделями на задаче безусловной генерации на наборе данных Wikipedia, длина генерации 128 токенов, нормализация DataNorm.

	PPL ↓	Mauve ↑	Div ↑	Mem ↓
(1)	71.9	0.908	0.447	0.355
(2)	71.6	0.914	0.527	0.304
LD4LG	70.3	0.912	0.533	0.295
TEncDM	68.9	0.921	0.540	0.286

Таблица 3: Результаты запуска конфигураций на безусловной задаче

4.3.2 Запуск на задаче суммаризации

Для сравнения выбран набор данных XSUM, длина генерации 40 токенов. В таблице 4 представлены результаты сравнения на задаче суммаризации.

	Rouge-1 ↑	Rouge-2 ↑	Rouge-L ↑	BERT Score ↑
(1)	0.227	0.040	0.175	0.599
(2)	0.237	0.052	0.182	0.642
LD4LG	0.381	0.159	0.312	0.748
TEncDM	0.337	0.119	0.271	0.698

Таблица 4: Результаты сравнения на задаче суммаризации

4.3.3 Анализ результатов

Как видно из таблицы 3, модель обученная во второй конфигурация по всем метрикам обходит модель обученную в первую конфигурации, это может быть связано с тем, что у второй конфигурации связь между диффузионной моделью и автокодировщиком сильнее, это может быть важно для задачи безусловной генерации. Также хочется отметить, что хоть обе конфигурации проигрывают существующим моделям, но значение метрик отличается всего на 5-10 процентов.

На задаче суммаризации таблица 4 ситуация такая же, снова конфигурация 2 показывает себя лучше, чем конфигурация 1. Также можно увидеть, что на этой задаче существующие диффузионные модели лучше в 1,5 - 2 раза по всем метрикам.

5 Будущая работа

Как видно из таблиц 3, 4 предоставленные конфигурации проигрывают по метрикам существующим моделям без применения одновременного обучение, это означает, что данные конфигурации еще требуют улучшения, данную задачу улучшения мы оставляем на будущее, так как в данной работе хочется показать возможность использования технологии одновременного обучения для диффузионных моделей.

6 Выводы

В данной работе было рассмотрено два подхода к одновременному обучению диффузионной модели и текстового автокодировщика. Первый подход основывался на обучении автокодировщика независимо от диффузии, хотя и с учётом её выходов. Во втором подходе была усилена связь между автокодировщиком и диффузионной моделью, добавив слагаемое в функцию потерь автокодировщика, учитывающее выход диффузии.

Проведённые эксперименты показали, что:

1. Непосредственное совместное обучение автокодировщика и диффузионной модели возможно и упрощает общий конвейер, так как не требуется отдельного этапа предварительного обучения автокодировщика.
2. Выбор нормализации и объёма обучающих данных существенно влияет на качество. На малых наборах данных модель склонна к переобучению и деградации качества. Расширение объёма обучающей выборки (например, с Rocstories на Wikipedia) положительно сказывается на результатах.
3. Использование DataNorm для нормализации выходов автокодировщика дает лучшие результаты по сравнению с другими рассмотренными способами нормализации (BatchNorm, LayerNorm, SphereNorm).
4. Вторая конфигурация, усиливающая связь между автокодировщиком и диффузионной моделью, в ряде случаев показывает более высокое качество на задачах безусловной генерации текста по сравнению с первой конфигурацией.
5. Несмотря на улучшения по сравнению с первой конфигурацией, предложенные подходы пока уступают по качеству существующим методам, в которых автокодировщик и диффузионная модель обучаются раздельно.

Таким образом, совместное обучение автокодировщика и диффузионной модели является перспективным направлением, позволяющим потенциально упростить и унифицировать процесс обучения. Однако для достижения конкурентного качества требуется дополнительная проработка архитектурных решений, функций потерь и методов нормализации, а также экспериментирование с ещё более крупными наборами данных и новыми методами регуляции.

Список литературы

1 Generative adversarial nets / Goodfellow Ian, Pouget-Abadie Jean, Mirza Mehdi, Xu Bing, Warde-Farley David, Ozair Sherjil, Courville Aaron, and Bengio Yoshua // Advances in neural information processing systems. — 2014. — Vol. 27.

2 Kingma Diederik P, Welling Max. Auto-encoding variational bayes // arXiv preprint arXiv:1312.6114. — 2013.

3 Rezende Danilo Jimenez, Mohamed Shakir, Wierstra Daan. Stochastic backpropagation and approximate inference in deep generative models // International conference on machine learning / PMLR. — 2014. — P. 1278–1286.

4 Ffjord: Free-form continuous dynamics for scalable reversible generative models / Grathwohl Will, Chen Ricky TQ, Bettencourt Jesse, Sutskever Ilya, and Duvenaud David // arXiv preprint arXiv:1810.01367. — 2018.

5 Residual flows for invertible generative modeling / Chen Ricky TQ, Behrmann Jens, Duvenaud David K, and Jacobsen Jörn-Henrik // Advances in Neural Information Processing Systems. — 2019. — Vol. 32.

6 Vaebm: A symbiosis between variational autoencoders and energy-based models / Xiao Zhisheng, Kreis Karsten, Kautz Jan, and Vahdat Arash // arXiv preprint arXiv:2010.00654. — 2020.

7 Deep unsupervised learning using nonequilibrium thermodynamics / Sohl-Dickstein Jascha, Weiss Eric, Maheswaranathan Niru, and Ganguli Surya // International Conference on Machine Learning / PMLR. — 2015. — P. 2256–2265.

8 Ho Jonathan, Jain Ajay, Abbeel Pieter. Denoising diffusion probabilistic models // arXiv preprint arXiv:2006.11239. — 2020.

9 Structured denoising diffusion models in discrete state-spaces / Austin Jacob, Johnson Daniel D, Ho Jonathan, Tarlow Daniel, and van den Berg Rianne // Advances in Neural Information Processing Systems. — 2021. — Vol. 34. — P. 17981–17993.

10 Diffusion-LM Improves Controllable Text Generation. — 2022. — 2205.14217.

11 DiffuSeq: Sequence to Sequence Text Generation with Diffusion Models. — 2023. — 2210.08933.

12 Empowering Diffusion Models on the Embedding Space for Text Generation / Gao Zhujin, Guo Junliang, Tan Xu, Zhu Yongxin, Zhang Fang, Bian Jiang, and Xu Linli // Proceedings of the 2024 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies (Volume 1: Long Papers) / ed. by Duh Kevin, Gomez Helena, Bethard Steven. — Mexico City, Mexico : Association for Computational Linguistics. — 2024. — June. — P. 4664–4683. — Access mode: https://aclanthology.org/2024.naacl-long.261.

13 PLANNER: Generating Diversified Paragraph via Latent Language Diffusion Model. — 2024. — 2306.02531.

14 DINOISER: Diffused Conditional Sequence Learning by Manipulating Noises. — 2024. — 2302.10025.

15 Arjovsky Martin, Chintala Soumith, Bottou Léon. Wasserstein gan. arXiv 2017 // arXiv preprint arXiv:1701.07875. — 2017. — Vol. 30. — P. 4.

16 Improved training of wasserstein gans / Gulrajani Ishaan, Ahmed Faruk, Arjovsky Martin, Dumoulin Vincent, and Courville Aaron C // Advances in neural information processing systems. — 2017. — Vol. 30.

17 Karras Tero, Laine Samuli, Aila Timo. A style-based generator architecture for generative adversarial networks // Proceedings of the IEEE/CVF conference on computer vision and pattern recognition. — 2019. — P. 4401–4410.

18 Alias-free generative adversarial networks / Karras Tero, Aittala Miika, Laine Samuli, Härkönen Erik, Hellsten Janne, Lehtinen Jaakko, and Aila Timo // Advances in Neural Information Processing Systems. — 2021. — Vol. 34.

19 Brock Andrew, Donahue Jeff, Simonyan Karen. Large scale GAN training for high fidelity natural image synthesis // arXiv preprint arXiv:1809.11096. — 2018.

20 Bias and generalization in deep generative models: An empirical study / Zhao Shengjia, Ren Hongyu, Yuan Arianna, Song Jiaming, Goodman Noah, and Ermon Stefano // Advances in Neural Information Processing Systems. — 2018. — Vol. 31.

21 Thanh-Tung Hoang, Tran Truyen. Catastrophic forgetting and mode collapse in GANs // 2020 International Joint Conference on Neural Networks (IJCNN) / IEEE. — 2020. — P. 1–10.

22 Song Yang, Ermon Stefano. Generative modeling by estimating gradients of the data distribution // arXiv preprint arXiv:1907.05600. — 2019.

23 Song Yang, Ermon Stefano. Improved techniques for training score-based generative models // arXiv preprint arXiv:2006.09011. — 2020.

24 Song Jiaming, Meng Chenlin, Ermon Stefano. Denoising diffusion implicit models // arXiv preprint arXiv:2010.02502. — 2020.

25 Score-based generative modeling through stochastic differential equations / Song Yang, Sohl-Dickstein Jascha, Kingma Diederik P, Kumar Abhishek, Ermon Stefano, and Poole Ben // arXiv preprint arXiv:2011.13456. — 2020.

26 Nichol Alex, Dhariwal Prafulla. Improved denoising diffusion probabilistic models // arXiv preprint arXiv:2102.09672. — 2021.

27 Dhariwal Prafulla, Nichol Alex. Diffusion models beat gans on image synthesis // arXiv preprint arXiv:2105.05233. — 2021.

28 Variational diffusion models / Kingma Diederik P, Salimans Tim, Poole Ben, and Ho Jonathan // arXiv preprint arXiv:2107.00630. — 2021. — Vol. 2.

29 Image super-resolution via iterative refinement / Saharia Chitwan, Ho Jonathan, Chan William, Salimans Tim, Fleet David J, and Norouzi Mohammad // arXiv preprint arXiv:2104.07636. — 2021.

30 Grad-tts: A diffusion probabilistic model for text-to-speech / Popov Vadim, Vovk Ivan, Gogoryan Vladimir, Sadekova Tasnima, and Kudinov Mikhail // International Conference on Machine Learning / PMLR. — 2021. — P. 8599–8608.

31 Liu Songxiang, Su Dan, Yu Dong. DiffGAN-TTS: High-Fidelity and Efficient Text-to-Speech with Denoising Diffusion GANs // arXiv preprint arXiv:2201.11972. — 2022.

32 Luo Shitong, Hu Wei. Diffusion probabilistic models for 3d point cloud generation // Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition. — 2021. — P. 2837–2845.

33 Zhou Linqi, Du Yilun, Wu Jiajun. 3d shape generation and completion through point-voxel diffusion // Proceedings of the IEEE/CVF International Conference on Computer Vision. — 2021. — P. 5826–5835.

34 A Theoretical Analysis of the Repetition Problem in Text Generation. — 2021. — 2012.14660.

35 Diffusion-LM improves controllable text generation / Li Xiang Lisa, Thickstun John, Gulrajani Ishaan, Liang Percy, and Hashimoto Tatsunori // arXiv preprint arXiv:2305.09515. — 2022.

36 Chen Ting, Zhang Ruixiang, Hinton Geoffrey. Analog bits: Generating discrete data using diffusion models with self-conditioning // arXiv preprint arXiv:2208.04202. — 2022.

37 SeqDiffSeq: Text diffusion with encoder-decoder transformers / Yuan Hongyi, Yuan Zhengyuan, Tan Chuanqi, Huang Fei, and Huang Songfang // Proceedings of the 2022 Conference on Empirical Methods in Natural Language Processing (EMNLP). — 2022.

38 Text Generation with Diffusion Language Models: A Pre-training Approach with Continuous Paragraph Denoise / Lin Zhenghao, Gong Yeyun, Shen Yelong, Wu Tong, Fan Zhihao, Lin Chen, Duan Nan, and Chen Weizhu // Proceedings of the 2023 Conference on Empirical Methods in Natural Language Processing (EMNLP). — 2023.

39 Han Xiaochuang, Kumar Sachin, Tsvetkov Yulia. SSD-LM: Semi-autoregressive Simplex-based Diffusion Language Model for Text Generation and Modular Control. — 2023. — 2210.17432.

40 TESS: Text-to-Text Self-Conditioned Simplex Diffusion. — 2024. — 2305.08379.

41 Documenting large webtext corpora: A case study on the colossal clean crawled corpus / Dodge Jesse, Sap Maarten, Marasović Ana, Agnew William, Ilharco Gabriel, Groeneveld Dirk, Mitchell Margaret, and Gardner Matt // arXiv preprint arXiv:2104.08758. — 2021.

42 LLaMA: Open and Efficient Foundation Language Models. — 2023. — 2302.13971.

43 Latent Diffusion for Language Generation. — 2023. — 2212.09462.

44 TEncDM: Understanding the Properties of the Diffusion Model in the Space of Language Model Encodings. — 2024. — 2402.19097.