
Исследование влияния модульной гибридизации на точность и эффективность трансформеров для долгосрочного прогнозирования временных рядов *

Очкин Н.В.
email@email

Зубарев К.М.
email@email

Аннотация

Задача долгосрочного прогнозирования многомерных временных рядов остается сложной: в реальных данных наблюдается ряд проблемных эффектов - среди них, краткосрочные мотивы (скачки, ступени), крайне долгосрочные и межрядовые зависимости, выраженная нестационарность (сдвиги тренда/уровня, мультисезонность, гетероскедастичность). В данной работе мы предлагаем расширить слой эмбединга в модели Informer [16] компактным двухслойным сверточным блоком с целью повышения эффективности извлечения локальных паттернов, внедрить модуль декомпозиции ряда из Autoformer [14] для явного разделения трендовых и сезонных компонент и заменить механизм ProbSparse на линейное внимание FAVOR+ (Performer [2]) для учёта глобальных зависимостей при низких вычислительных затратах. Эксперименты на стандартных бенчмарках демонстрируют снижение ошибок на длинных горизонтах прогнозирования в ряде настроек. Абляционные исследования указывают на вклад каждого из модулей, хотя эффект не универсален для всех горизонтов. На рассмотренных конфигурациях наблюдается благоприятный компромисс между точностью и затратами времени/памяти.

Keywords LSTF · Long Sequence Time-series Forecasting · Transformer · Informer · Performer · Autoformer

1 Введение

Долгосрочное прогнозирование временных рядов является одной из ключевых задач во многих областях - от энергетики и финансов до транспорта и здравоохранения. Однако реальные многомерные ряды редко обладают простой структурой: среди прочего, они сочетают краткосрочные локальные паттерны, очень длинные и межрядовые зависимости, а также выраженную нестационарность, включая сдвиги тренда, мультисезонность и гетероскедастичность. Эти свойства затрудняют обучение моделей и ухудшают их способность к экстраполяции на длительных горизонтах.

С момента публикации, архитектура Трансформера [13] завоевала широкое признание. Однако у нее есть несколько серьезных проблем, которые усложняют работу с длинными временными последовательностями (LSTF). Последующие исследования предложили различные методы решения данных и связанных с ними проблем (Informer [16], Autoformer [14], Performer [2]). Тем не менее, существующие архитектуры сталкиваются с рядом ограничений: необходимость одновременного учёта локальных и глобальных закономерностей, высокая чувствительность к нестационарности и ограниченная масштабируемость по длине входной последовательности.

* Исходный код, конфигурации и скрипты обучения доступны: <https://github.com/namenick91/Convformer>

В данной работе мы предлагаем архитектуру, которая объединяет три взаимодополняющих индуктивных смещения, специально ориентированных на решение этих вызовов: **(i)** расширенный сверточный входной блок, способный эффективно кодировать краткосрочные локальные паттерны (скачки, ступени, импульсные всплески). Такой модуль выполняет роль фильтра низкого уровня и одновременно стабилизирует статистики входных данных, что повышает устойчивость модели к локальной нестационарности; **(ii)** механизм внимания FAVOR+ (Performer [2]), обеспечивающий линейные $O(Lr)$ затраты времени и памяти; параметр ранга r задаёт настраиваемый компромисс между скоростью и точностью. Это позволяет обрабатывать очень длинные последовательности и улавливать отложенные зависимости без взрыва затрат, что критично для прогнозов на больших горизонтах. **(iii)** явная декомпозиция ряда в стиле Autoformer [14] после каждого блока self-attention: низкочастотный тренд выводится через остаточные соединения, а очищенный от тренда стационаризованный остаток подаётся в механизм внимания. Такой приём снижает влияние сдвигов уровня и многосезонности, улучшая обобщающую способность модели.

Такая композиция локальной фильтрации, глобального внимания и явной декомпозиции улучшает точность прогнозирования на длинных горизонтах, ускоряет и стабилизирует оптимизацию, а также снижает затраты памяти по сравнению с базовыми трансформер-подходами. Основной вклад данной работы можно резюмировать следующим образом:

- Вводим расширенный сверточный входной блок для кодирования краткосрочных мотивов и стабилизации статистик нестационарных входов.
- Для моделирования глобальных связей применяем FAVOR+ внимание с линейными затратами по времени и памяти, что обеспечивает масштабируемое и не зависящее от распределения моделирование глобальных зависимостей;
- Встраиваем Autoformer-подобную [14] декомпозицию после каждого блока self-attention, которая пропускает низкочастотный тренд по остаточным соединениям и передаёт более стационаризованный поток данных в механизм внимания.
- Эксперименты на стандартных бенчмарках демонстрируют устойчивое снижение ошибок на длительных горизонтах прогнозирования; абляционные исследования подтверждают вклад каждого модуля, а анализ масштабируемости показывает благоприятный баланс между точностью и вычислительными затратами.

2 Обозначения и предварительные сведения

Приведём формальное определение задачи долгосрочного прогнозирования временных рядов (LSTF). Рассматривается дискретный многомерный временной ряд $\{\mathbf{x}_t\}_{t=1}^T$, $\mathbf{x}_t \in \mathbb{R}^{C_{\text{in}}}$. Пусть $\mathbf{y}_t \in \mathbb{R}^{C_{\text{out}}}$ обозначает вектор целевых компонент \mathbf{x}_t , причём $C_{\text{out}} \leq C_{\text{in}}$. Для каждого момента времени t , удовлетворяющего $L_x \leq t \leq T - L_y$, определим входное окно наблюдений $\mathcal{X}_t = \{\mathbf{x}_{t-L_x+1}, \dots, \mathbf{x}_t\}$, и соответствующую ему последовательность будущих целевых значений $\mathcal{Y}_t = \{\mathbf{y}_{t+1}, \dots, \mathbf{y}_{t+L_y}\}$. Здесь и в дальнейшем: t - глобальный индекс времени, L_x, L_y - длины входной и выходной последовательностей, $C_{\text{in}}, C_{\text{out}}$ - число входных и прогнозируемых признаков (измеряемых величин) в каждый момент времени соответственно. Дальнейшие рассуждения одинаково применимы к многомерному и одномерному случаям; различие заключается лишь в выборе C_{out} и в том, какие компоненты \mathbf{x}_t считаются таргетами.

Задача LSTF формулируется как задача обучения параметризованного отображения $f_\theta : \mathbb{R}^{L_x \times C_{\text{in}}} \rightarrow \mathbb{R}^{L_y \times C_{\text{out}}}$, которое по входному окну наблюдений \mathcal{X}_t восстанавливает соответствующую ему последовательность будущих значений \mathcal{Y}_t :

$$\hat{\mathcal{Y}}_t = f_\theta(\mathcal{X}_t) \approx \mathcal{Y}_t.$$

Задача LSTF предполагает предсказание далёкого будущего, то есть больших значений L_y ; при этом размерность признаков не ограничивается одномерным случаем. Пусть обучающая выборка состоит из набора окон $\mathcal{D} = \{(\mathcal{X}_t, \mathcal{Y}_t)\}_{t \in \mathcal{T}}$, где \mathcal{T} - множество индексов времени, для которых формируются пары $(\mathcal{X}_t, \mathcal{Y}_t)$. Обучение модели f_θ проводится в постановке контролируемого обучения путём минимизации среднеквадратичной функции потерь:

$$\mathcal{L}(\theta) = \frac{1}{|\mathcal{T}|} \sum_{t \in \mathcal{T}} \frac{1}{L_y \cdot C_{\text{out}}} \|f_\theta(\mathcal{X}_t) - \mathcal{Y}_t\|_F^2,$$

где $\|\cdot\|_F$ обозначает норму Фробениуса. В дальнейшем под f_θ рассматриваются как базовые трансформерные архитектуры Informer, Performer и Autoformer, так и предлагаемые в данной работе модификации. Все модели обучаются в единой постановке, отличаясь лишь внутренней реализацией отображения f_θ при фиксированных $L_x, L_y, C_{in}, C_{out}$.

2.1 Базовые компоненты архитектуры

Классический механизм внимания. дописать multi-head attention? Пусть $\mathbf{Q} \in \mathbb{R}^{L_Q \times d_k}$, $\mathbf{K} \in \mathbb{R}^{L_K \times d_k}$, $\mathbf{V} \in \mathbb{R}^{L_V \times d_v}$ – промежуточные представления входных данных, строки которых можно интерпретировать как запросы, ключи и значения непрерывной словарной структуры данных соответственно [13], где L_Q и L_K обозначают длину последовательностей запросов и ключей/значений соответственно, d_k – размерность запросов и ключей, а d_v – размерность значений. Двухнаправленное (bidirectional, или неориентированное (non-directional) [3]) внимание на основе скалярного произведения имеет вид:

$$\text{Att}_{\leftrightarrow}(\mathbf{Q}, \mathbf{K}, \mathbf{V}) = \text{softmax}\left(\frac{\mathbf{Q}\mathbf{K}^\top}{\sqrt{d_k}}\right) \mathbf{V} \in \mathbb{R}^{L_Q \times d_v}, \quad (1)$$

где $\text{softmax}(\cdot)$ применяется построчно.

Временная и пространственная сложность вычисления (1) равны $O(L_Q L_K (d_k + d_v))$ и $O(L_Q L_K + L_Q d_k + L_K d_k + L_K d_v)$ соответственно (при $L_Q = L_K = L$ и $d_v = d_k$ получаем $O(L^2 d_k)$ по времени и $O(L^2 + L d_k)$ по памяти). Поэтому механизм внимания на основе скалярного произведения (1) в принципе несовместим с end-to-end-обработкой длинных последовательностей. Двухнаправленное внимание используется в self-attention энкодера и в cross-attention энкодер–декодера в архитектурах Seq2Seq.

Другой важный тип внимания – однонаправленное (unidirectional) внимание:

$$\text{Att}_{\rightarrow}(\mathbf{Q}, \mathbf{K}, \mathbf{V}) = \text{softmax}\left(\frac{\mathbf{Q}\mathbf{K}^\top}{\sqrt{d_k}} + \mathbf{M}\right) \mathbf{V} \in \mathbb{R}^{L_Q \times d_v},$$

где $\mathbf{M} \in \mathbb{R}^{L_Q \times L_K}$ – маска, элементы которой задают, какие логиты участвуют в нормализации (элементы, равные $-\infty$, полностью подавляют соответствующие ключи для данного запроса).

Однонаправленное внимание используется в self-attention декодера в архитектурах Seq2Seq. В частности, в случае self-attention декодера, когда $L_Q = L_K = L$, обычно используют нижнетреугольную каузальную маску, задаваемую правилом $M_{ij} = 0$ при $j \leq i$ и $M_{ij} = -\infty$ при $j > i$.

Аппроксимация softmax-ядер для механизма внимания. TODO

Декомпозиция ряда. Следуя Autoformer [14], опишем механизм декомпозиции ряда (SeriesDecomp) – внутреннюю операцию, которая постепенно извлекает долгосрочную трендовую (трендово-циклическую) составляющую из предсказанных промежуточных скрытых представлений. Формально:

$$\begin{aligned} \mathcal{X}_t^{(\text{tr})} &= \text{MA}_k(\mathcal{X}_t), \\ \mathcal{X}_t^{(\text{se})} &= \mathcal{X}_t - \mathcal{X}_t^{(\text{tr})}, \end{aligned}$$

где $\mathcal{X}_t^{(\text{tr})}, \mathcal{X}_t^{(\text{se})} \in \mathbb{R}^{L_x \times d_{\text{model}}}$ – трендово-циклическая и сезонная составляющие соответственно, $\text{MA}_k(\cdot)$ – оператор скользящего среднего (simple moving average) с окном длины k , единичным шагом и постоянным дополнением на границах. В дальнейшем будем использовать обозначение $(\mathcal{X}_t^{(\text{tr})}, \mathcal{X}_t^{(\text{se})}) = \text{SeriesDecomp}(\mathcal{X}_t)$ для краткой записи описанного выше блока.

Представление входных данных. TODO

Дистилляция внимания. TODO

3 Методология

Повышение эффективности извлечения локальных паттернов Для усиления способности модели к распознаванию краткосрочных закономерностей мы предлагаем заменить стандартный механизм представления

значений в блоке с обработкой входных данных в Informer [16] на сверточный блок ConvStem, при этом оставив позиционное с темпоральным эмбеддированием. Данный блок сочетает проекцию входных признаков через точечную свёртку с последующими операциями широкой и глубокой свёрток, дополненных нормализацией и нелинейностью. Такое построение позволяет модели фиксировать повторяющиеся локальные мотивы и вариации формы сигналов непосредственно на этапе встраивания, ещё до применения механизмов внимания. В результате глобальное внимание может быть сфокусировано преимущественно на долгосрочных зависимостях, тогда как локальная динамика эффективно извлекается специализированным сверточным модулем.

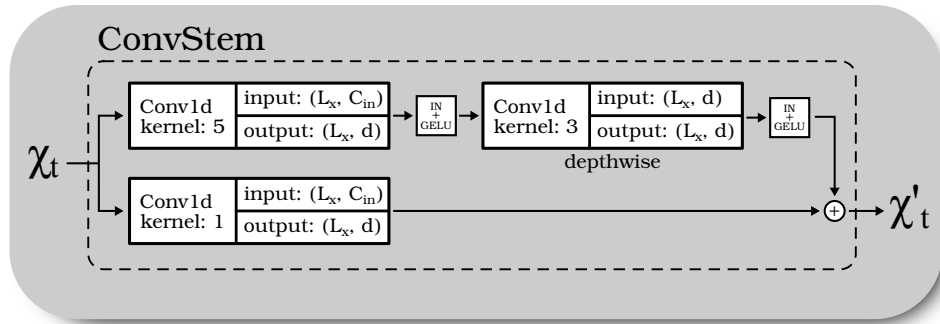


Рис. 3.1: Схема блока ConvStem. Здесь: $X_t \in \mathbb{R}^{L_x \times C_{in}}$, $X'_t \in \mathbb{R}^{L_x \times d}$, где $d = d_{model}$ – размерность модели; IN – нормализация по экземплярам (Instance Normalization [12]). Псевдокод приведен в 1.

Масштабируемое моделирование глобальных зависимостей В качестве механизма внимания предлагается использовать FAVOR+ (Performer [2]) вместо ProbSparse (оригинально применяемого в Informer [16]). Подобно тому, как в оригинальном Informer [16] механизм FullAttention [13] был заменен на ProbSparse исключительно для вычисления self-attention (механизм самовнимания) в слоях кодировщика и декодера, в нашей модификации ProbSparse заменяется на FAVOR+ в тех же местах, тогда как cross-attention останется реализованным через полное внимание (FullAttention). Данную замену мы мотивируем тем, что FAVOR+ обеспечивает несмещённую аппроксимацию softmax-ядра с линейной по времени и памяти сложностью [2], что гарантирует предсказуемую и масштабируемую работу на очень длинных последовательностях в условиях ограниченных ресурсов GPU. При этом точность аппроксимации управляется числом случайных признаков r : при меньших значениях достигается высокая скорость, при больших - более точное восстановление распределения внимания. Такая настраиваемость делает механизм универсальным инструментом, позволяющим варьировать баланс между эффективностью и качеством. Кроме того, отказ от ручной выборки top- u запросов позволяет модели учитывать любые глобальные зависимости, а не только наиболее сильные периодические, что критично для учёта неперiodических скачков в реальных временных рядах.

Явное разделение тренда и сезонности Чтобы явно разделить тренд и сезонную компоненту, мы интегрировали механизм декомпозиции временных рядов, заимствованный из архитектуры Autoformer [14]. После каждого блока self-attention скрытое представление разлагается на два канала: низкочастотный тренд и остаточную компоненту, содержащую более стационарные сезонные и случайные колебания. Тренд передаётся по остаточному пути, обеспечивая стабильность прогнозов при сдвигах уровня, тогда как внимание применяется к сезонной компоненте, где наиболее выражены периодические и локальные закономерности. Такая стратегия снижает вариативность обучения, повышает способность к экстраполяции за пределами обучающего диапазона и улучшает интерпретируемость результатов за счёт явного выделения трендовой и сезонной составляющих.

Input: X

(1) [SeriesDecomp] \rightarrow Trend, Seasonal

Trend = AvgPool(Padding(X))
where Seasonal = $X - \text{Trend}$

(2) [self-attention mechanism] on Seasonal

\rightarrow Seasonal' (+ residual connection, normalization, etc.)

(3) [Feed-Forward] on Seasonal' \rightarrow $\widetilde{\text{Seasonal}}$

(4) Re-compose $X = \widetilde{\text{Seasonal}} + \text{Trend}$

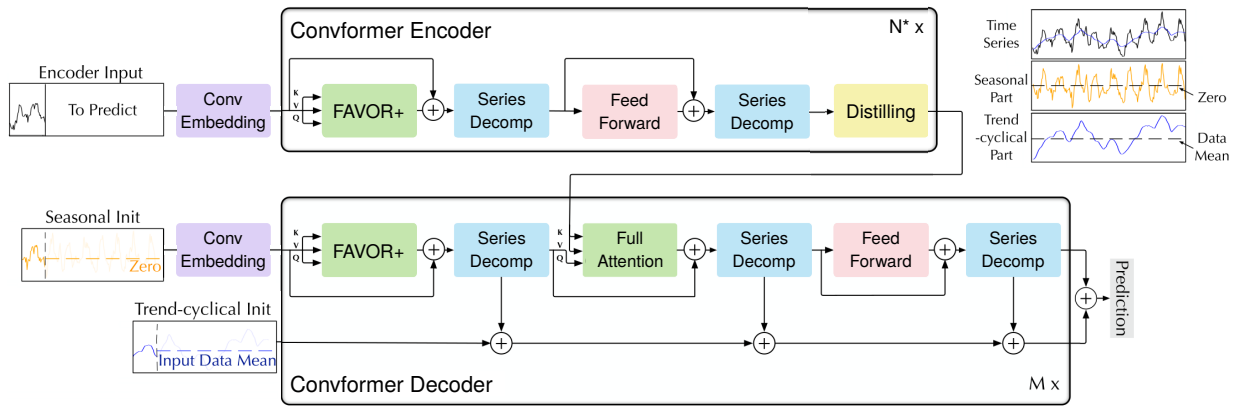


Рис. 3.2: Архитектура Convformer. Входные данные (для декодера – их сезонная составляющая) проходят через блок ConvEmbedding (фиолетовый блок), состоящий из ConvStem, PositionalEmbedding и TemporalEmbedding (подробнее см. [ДОБАВИТЬ ССЫЛКУ НА АППЕНДИКС](#)). Энкодер устраняет долгосрочную трендово-циклическую составляющую с помощью блоков декомпозиции рядов (синие блоки) и сосредотачивается на моделировании сезонных компонент. Блок дистилляции N^* в энкодере (жёлтый блок) включён во все слои, кроме последнего. Декодер постепенно накапливает трендовую составляющую, извлекаемую из скрытых переменных. Прошлая сезонная информация, полученная от энкодера, используется в классическом (полном) механизме внимания ([ССЫЛКА НА БЛОК С FULL ATTENTION](#)) (центральный зелёный блок в декодере).

В итоге архитектура объединяет сильные стороны трёх подходов: локальное кодирование паттернов через ConvStem, линейное глобальное внимание FAVOR+ и явную декомпозицию ряда из Autoformer [14], сохраняя при этом полную совместимость с остальными гиперпараметрами оригинального Informer [16].

4 Эксперимент

Датасет Ниже приводится описание набора данных экспериментов: *ETT* [16] датасет содержит информацию, собранную с электрических трансформаторов, включая нагрузку и температуру масла, которые регистрировались каждые 15 минут в период с июля 2016 года по июль 2018 года.

Детали реализации Обучение проводилось с использованием функции потерь MSE (L2) и оптимизатора Adam [9] с начальной скоростью обучения 10^{-4} . Размер батча - 32. Процесс обучения досрочно останавливается в пределах 10 эпох. Результаты экспериментов были получены в ходе усреднения по трем отдельным запускам, реализованы в PyTorch [11] и проводились на одной графической карте NVIDIA GTX 1660 SUPER с 6 ГБ видеопамяти. Архитектура модели включает 2 слоя энкодера и 1 слой декодера. Полный перечень гиперпараметров и конфигураций приведён в репозитории: <https://github.com/namenick91/Convformer>

Horizon		Convformer			Informer			Performer			Autoformer		
		MSE	MAE	t	MSE	MAE	t	MSE	MAE	t	MSE	MAE	t
ETTh1	24	0.388	0.428	3m50s/4s	0.524	0.527	2m5s/3s	0.598	0.570	2m50s/4s	0.401	0.425	3m58s/7s
	48	0.435	0.451	3m55s/4s	0.631	0.601	2m24s/4s	0.765	0.673	2m35s/4s	0.430	0.445	4m43s/7s
	168	0.435	0.459	8m39s/6s	0.825	0.705	3m14s/5s	0.918	0.768	3m42s/6s	0.478	0.473	6m45s/11s
	336	0.469	0.490	7m55s/8s	1.310	0.937	9m19s/6s	1.024	0.823	5m36s/7s	0.516	0.497	10m19s/17s
	720	0.510	0.528	12m51s/10s	1.205	0.879	14m36s/9s	1.107	0.843	14m9s/9s	—	—	—
ETT2	24	0.248	0.345	2m38s/4s	1.284	0.891	1m58s/3s	1.296	0.907	2m49s/4s	0.290	0.365	4m7s/6s
	48	0.298	0.373	2m49s/5s	1.559	1.008	1m58s/4s	1.568	1.008	2m35s/4s	0.324	0.382	3m31s/6s
	168	0.539	0.509	4m6s/6s	7.587	2.335	2m49s/5s	8.487	2.569	3m40s/6s	0.451	0.456	5m15s/9s
	336	0.684	0.600	7m9s/8s	4.369	1.773	4m19s/7s	8.158	2.456	6m16s/7s	0.478	0.480	7m3s/13s
	720	0.662	0.595	13m1s/11s	2.977	1.467	6m1s/9s	3.707	1.640	7m39s/9s	—	—	—

Таблица 1: Результаты многомерных предсказаний на 2 датасетах ETT с горизонтами предсказаний: { 24, 48, 168, 336, 720 }. Мы фиксируем входную длину последовательностей у моделей как 96. Символ “—” обозначает выход за пределы памяти (OOM). Время указано как train/inference. Полный бенчмарк см. в табл. ?? . Примеры визуализации приведены на рис. B.1.

4.1 Абляционные исследования

ConvStem вместо TokenEmbedding Сравнивалась эффективность оригинальной модели Informer [16] и её модификации с расширенным слоем эмбединга 2.

Horizon	ConvStem			Informer		
	MSE	MAE	t	MSE	MAE	t
24	0.469	0.480	2m11s/4s	0.524	0.527	2m5s/3s
48	0.587	0.558	2m23s/4s	0.631	0.601	2m24s/4s
168	0.861	0.733	3m34s/5s	0.825	0.705	3m14s/5s
336	1.094	0.843	7m22s/7s	1.310	0.937	9m19s/6s
720	1.261	0.913	9m58s/9s	1.205	0.879	14m36s/9s

Таблица 2: Результаты многомерных предсказаний на датасете ETTh1 с горизонтами предсказаний: { 24, 48, 168, 336, 720 }. Входная длина последовательности фиксирована: 96. Время указано как train/inference. Примеры визуализации приведены в табл. 6.

Встраивание компактного сверточного блока на вход приводит к заметному улучшению качества на коротких (24-48) и особенно длинных горизонтах (336), при этом затраты по времени остаются сопоставимыми. Однако на отдельных горизонтах (например, 168) Informer [16] сохраняет преимущество. Это подтверждает, что сверточная фильтрация локальных паттернов в среднем повышает устойчивость к краткосрочной нестационарности, хотя её вклад не универсален.

ProbSparse → FAVOR+ Сравнивалась эффективность Informer [16] с исходным механизмом ProbSparse и модифицированной версии, в которой ProbSparse заменён на FAVOR+ (Performer [2]).

Horizon	Informer			Informer w/ FAVOR+		
	MSE	MAE	t	MSE	MAE	t
24	0.524	0.527	2m5s/3s	0.494	0.505	2m30s/4s
48	0.631	0.601	2m24s/4s	0.710	0.640	2m20s/4s
168	0.825	0.705	3m14s/5s	0.864	0.746	3m24s/5s
336	1.310	0.937	9m19s/6s	1.088	0.846	6m22s/7s
720	1.205	0.879	14m36s/9s	1.065	0.831	9m54s/9s

Таблица 3: Результаты многомерных предсказаний на датасете ETTh1 с горизонтами предсказаний: { 24, 48, 168, 336, 720 }. Входная длина последовательности фиксирована: 96. Время указано как train/inference. Примеры визуализации приведены в табл. 7.

Линейное внимание обеспечивает выигрыш по качеству на длинных горизонтах (336, 720) при сопоставимых или меньших вычислительных затратах, в то время как на коротких горизонтах улучшение выражено слабее или отсутствует. Таким образом, FAVOR+ [2] даёт наибольший эффект именно в режимах, где глобальные зависимости становятся критичными, подтверждая его ценность для масштабируемого долгосрочного прогнозирования.

Informer с модулем декомпозиции Сравнивалась эффективность оригинальной модели Informer [16] и модификации, дополненной механизмом декомпозиции ряда из Autoformer [14].

Horizon		Informer			Informer w/ s.decomp		
		MSE	MAE	t	MSE	MAE	t
ETTh1	24	0.524	0.527	2m5s/3s	0.478	0.494	3m36s/4s
	48	0.631	0.601	2m24s/4s	0.561	0.547	3m4s/4s
	168	0.825	0.705	3m14s/5s	0.649	0.597	7m2s/5s
	336	1.310	0.937	9m19s/6s	0.952	0.748	10m55s/7s
	720	1.205	0.879	14m36s/9s	1.059	0.775	16m46s/10s

Таблица 4: Результаты многомерных предсказаний на датасете ETTh1 с горизонтами предсказаний: { 24, 48, 168, 336, 720 }. Входная длина последовательности фиксирована: 96. Время указано как train/inference. Примеры визуализации приведены в табл. 8.

Встраивание операции разделения на тренд и остаток после каждого блока self-attention приводит к систематическому снижению ошибок на всех горизонтах ETTh1, хотя ценой становится рост времени обучения и инференса (inference). Это подтверждает гипотезу о том, что явная стабилизация нестационарности улучшает обобщающую способность модели, особенно при наличии сдвигов уровня и мультисезонности.

5 Заключение

Предложенная комбинация ConvStem + FAVOR+ + Autoformer-подобная декомпозиция демонстрирует улучшение качества на длинных горизонтах при сопоставимых вычислительных затратах на рассматриваемых настройках. Абляционные исследования подтверждают вклад модулей на большинстве горизонтов: ConvStem - выделение локальных мотивов; FAVOR+ - масштабируемые дальние зависимости; декомпозиция - устойчивость к тренду/сезонности. На ETT наибольшие выигрыши достигаются при длительных горизонтах (336-720), тогда как на средних горизонтах Autoformer остается сопоставимым.

При этом исследование имеет ряд ограничений: в качестве бенчмарка был рассмотрен один набор данных (ETT), cross-attention сохраняет квадратичную сложность, чувствительность MAPE/MSPE к масштабам признаков, отсутствие оценки неопределённости. Перспективными направлениями являются линейаризация cross-attention, адаптивный выбор ранга r в FAVOR+, расширение набора датасетов, включение калибровки и методов, устойчивых к дрейфу распределений.

СПИСОК ИСПОЛЬЗОВАННЫХ ИСТОЧНИКОВ

- [1] Dzmitry Bahdanau, Kyunghyun Cho, and Yoshua Bengio. Neural machine translation by jointly learning to align and translate. In International Conference on Learning Representations (ICLR), 2015.
- [2] Krzysztof Choromanski, Valentin Likhoshesterov, David Dohan, Xingyou Song, Alex Gane, Tamas Sarlos, Peter Hawkins, Jared Davis, Afroz Mohiuddin, Łukasz Kaiser, David Belanger, Lucy Colwell, and Adrian Weller. Rethinking attention with performers. In Proceedings of the International Conference on Learning Representations (ICLR), 2020. arXiv:2009.14794.
- [3] Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. BERT: Pre-training of deep bidirectional transformers for language understanding. CoRR, abs/1810.04805, 2018.
- [4] Gregory C. Reinsel George E. P. Box, Gwilym M. Jenkins and Greta M. Ljung. Time Series Analysis: Forecasting and Control. Wiley, fifth edition edition, 2015.
- [5] Ian Goodfellow, Yoshua Bengio, and Aaron Courville. Deep Learning. MIT Press, 2016. <http://www.deeplearningbook.org>.
- [6] James D. Hamilton. Time Series Analysis. Princeton University Press, 1994.
- [7] Rob J. Hyndman and George Athanasopoulos. Forecasting: principles and practice. OTexts, 2013.
- [8] Brockwell P. J. and Davis R. A. Introduction to Time Series and Forecasting. Springer, third edition edition, 2016.
- [9] Diederik P. Kingma and Jimmy Ba. Adam: A method for stochastic optimization. International Conference on Learning Representations (ICLR), 2015.
- [10] Kevin P. Murphy. Probabilistic Machine Learning: An introduction. MIT Press, 2022.
- [11] Adam Paszke, Sam Gross, Francisco Massa, Adam Lerer, James Bradbury, Gregory Chanan, Trevor Killeen, Zeming Lin, Natalia Gimelshein, Luca Antiga, Alban Desmaison, Andreas Köpf, Edward Yang, Zach DeVito, Martin Raison, Alykhan Tejani, Sasank Chilamkurthy, Benoit Steiner, Lu Fang, Junjie Bai, and Soumith Chintala. Pytorch: An imperative style, high-performance deep learning library. In Advances in Neural Information Processing Systems (NeurIPS), 2019.
- [12] Dmitry Ulyanov, Andrea Vedaldi, and Victor Lempitsky. Instance normalization: The missing ingredient for fast stylization. 2016.
- [13] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N. Gomez, Łukasz Kaiser, and Illia Polosukhin. Attention is all you need. In Advances in Neural Information Processing Systems (NeurIPS), volume 30, 2017.
- [14] Haixu Wu, Jiehui Xu, Jianmin Wang, and Mingsheng Long. Autoformer: Decomposition transformers with Auto-Correlation for long-term series forecasting. In Advances in Neural Information Processing Systems, 2021.
- [15] Haoyi Zhou, Jianxin Li, Shanghang Zhang, Shuai Zhang, Mengyi Yan, and Hui Xiong. Expanding the prediction capacity in long sequence time-series forecasting. Artificial Intelligence, 318:103886, 2023.
- [16] Haoyi Zhou, Shanghang Zhang, Jieqi Peng, Shuai Zhang, Jianxin Li, Hui Xiong, and Wancai Zhang. Informer: Beyond efficient transformer for long sequence time-series forecasting. In The Thirty-Fifth AAAI Conference on Artificial Intelligence, AAAI 2021, Virtual Conference, volume 35, pages 11106–11115. AAAI Press, 2021.

А Полный бенчмарк

Models		Convformer		Informer		Performer		Autoformer	
Metrics		MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE
ETTh1	24	0.388	0.428	0.524	0.527	0.598	0.570	0.401	0.425
	48	0.435	0.451	0.631	0.601	0.765	0.673	0.430	0.445
	168	0.435	0.459	0.825	0.705	0.918	0.768	0.478	0.473
	336	0.469	0.490	1.310	0.937	1.024	0.823	0.516	0.497
	720	0.510	0.528	1.205	0.879	1.107	0.843	–	–
ETTh2	24	0.248	0.345	1.284	0.891	1.296	0.907	0.290	0.365
	48	0.298	0.373	1.559	1.008	1.568	1.008	0.324	0.382
	168	0.539	0.509	7.587	2.335	8.487	2.569	0.451	0.456
	336	0.684	0.600	4.369	1.773	8.158	2.456	0.478	0.480
	720	0.662	0.595	2.977	1.467	3.707	1.640	–	–
ECL	24	0.161	0.282	0.253	0.356	0.265	0.371	0.170	0.290
	48	0.176	0.293	0.273	0.366	0.276	0.373	0.182	0.298
	168	0.198	0.312	0.288	0.381	0.286	0.383	0.220	0.329
	336	0.217	0.330	0.315	0.404	0.301	0.394	0.261	0.359
	720	–	–	–	–	–	–	–	–
Exchange	24	0.078	0.213	0.420	0.523	0.272	0.418	0.065	0.185
	48	0.118	0.257	0.584	0.614	0.437	0.520	0.119	0.253
	168	0.493	0.501	1.142	0.873	1.352	0.904	0.267	0.378
	336	0.621	0.581	1.767	1.061	1.892	1.071	0.468	0.508
	720	0.591	0.597	2.834	1.404	2.783	1.405	–	–
Illness	24	3.021	1.158	5.451	1.615	5.130	1.566	3.645	1.353
	36	2.962	1.147	5.059	1.541	4.904	1.502	3.511	1.300
	48	3.040	1.145	5.028	1.536	4.733	1.445	3.149	1.197
	60	2.921	1.113	5.340	1.578	5.212	1.533	2.905	1.152
	720	–	–	–	–	–	–	–	–
Traffic	24	0.533	0.341	0.705	0.398	0.667	0.382	0.587	0.388
	48	0.556	0.352	0.685	0.380	0.666	0.373	0.606	0.385
	168	0.597	0.378	0.740	0.404	0.669	0.368	–	–
	336	0.620	0.386	0.770	0.420	0.678	0.372	–	–
Weather	24	0.124	0.204	0.155	0.237	0.170	0.255	0.164	0.245
	48	0.167	0.248	0.240	0.323	0.320	0.392	0.233	0.309
	168	0.271	0.347	0.490	0.497	0.538	0.521	0.294	0.354
	336	0.425	0.460	0.626	0.554	0.657	0.577	0.354	0.389
	720	0.652	0.596	0.912	0.687	0.925	0.701	–	–
Count		45		0		2		17	

Таблица 5: Comparison of forecasting models across multiple datasets (MSE and MAE). **Note:** input length and predicted feature count vary by dataset.

В Дополнение к основным результатам

В.1 Визуализация абляционных исследований

Для оценки предсказаний различных моделей приводится визуализация результатов прогнозирования на тестовой выборке набора данных ETTh1 в рамках абляционных исследований, что позволяет провести качественное сравнение.

Таблица 6: Примеры прогнозирования из набора данных ETTh1 в ходе абляционных исследований *ConvStem* вместо *TokenEmbedding* (см. раздел 4.1) в настройках `input-96-predict-horizon`. Синие линии - истинные значения, оранжевые линии - предсказания модели, зеленые - входные данные длиной 96.

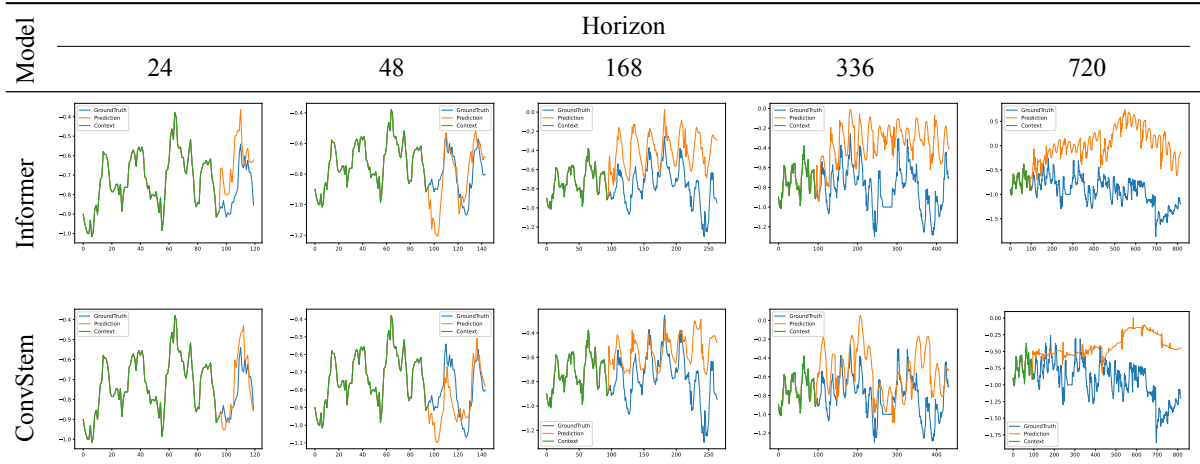


Таблица 7: Примеры прогнозирования из набора данных ETTh1 в ходе абляционных исследований *ProbSparse* \rightarrow *FAVOR+* (см. раздел 4.1) в настройках `input-96-predict-horizon`.

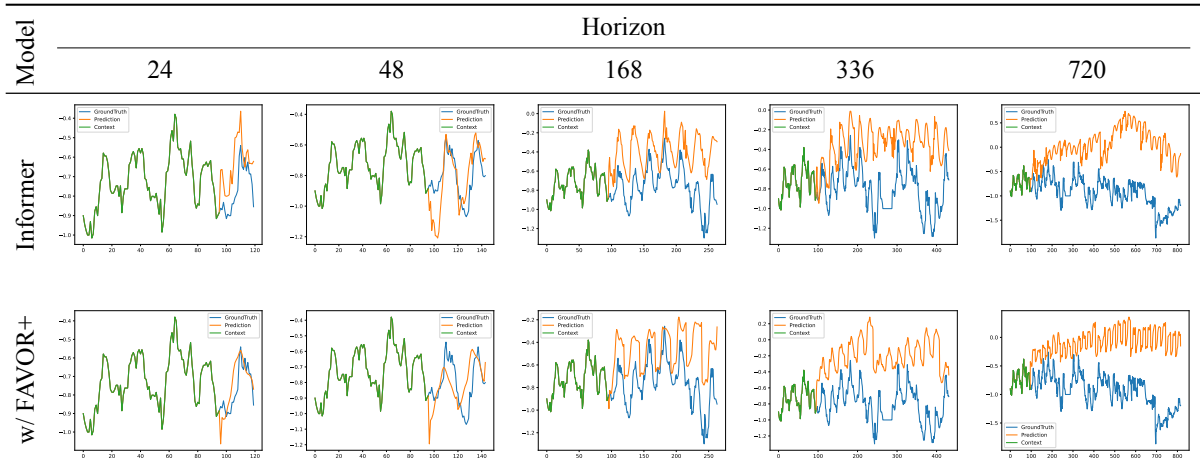
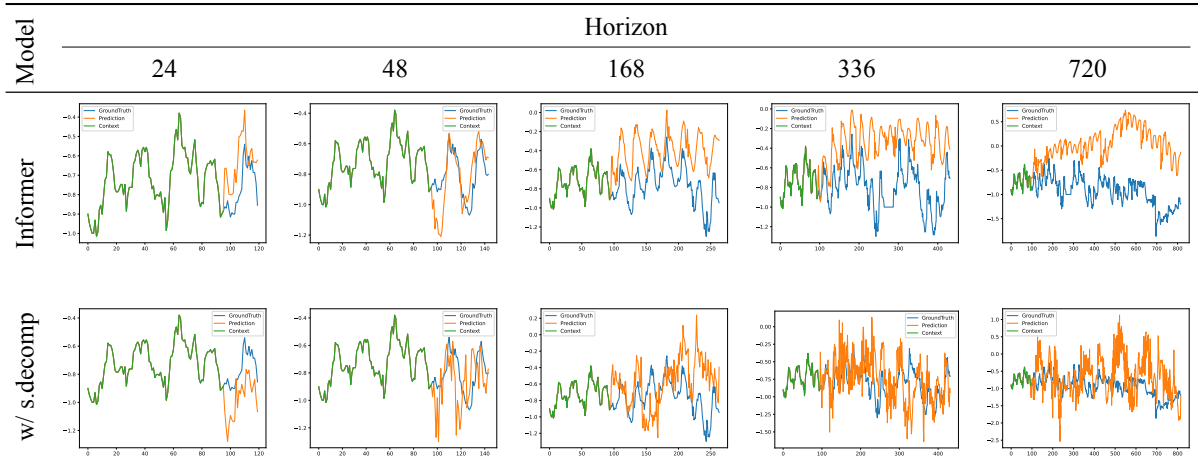


Таблица 8: Примеры прогнозирования из набора данных ETTh1 в ходе абляционных исследований *Informer* с модулем декомпозиции (см. раздел 4.1) в настройках `input-96-predict-horizon`.



B.2 Визуализация основных результатов

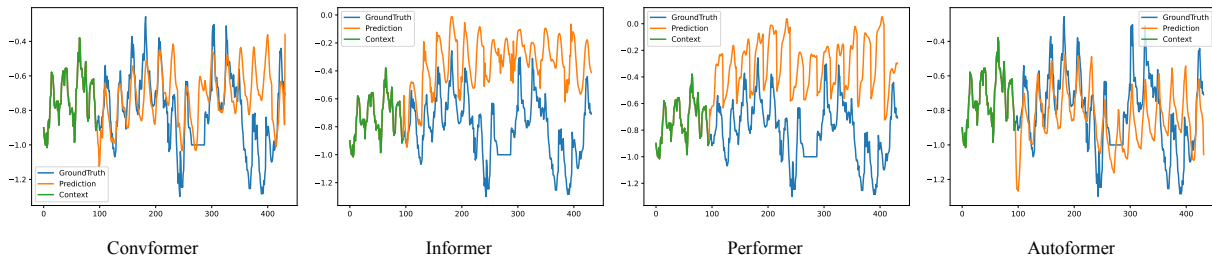


Рис. B.1: Примеры прогнозирования из набора данных ETTh1 в настройках `input-96-predict-336`.

С Детали реализации

Algorithm 1 Блок ConvStem

Require: входное окно $\mathcal{X}_t \in \mathbb{R}^{B \times L_x \times C_{\text{in}}}$

Ensure: выход $\mathcal{X}'_t \in \mathbb{R}^{B \times L_x \times d_{\text{model}}}$

1: $\mathcal{X}_t \leftarrow \text{Permute}(\mathcal{X}_t)$	$\triangleright \mathcal{X}_t \in \mathbb{R}^{B \times C_{\text{in}} \times L_x}$
2: $R \leftarrow \text{Conv1D}_{k=1, p=0}(\mathcal{X}_t)$	$\triangleright R \in \mathbb{R}^{B \times d_{\text{model}} \times L_x}$
3: $H_1 \leftarrow \text{GELU}(\text{IN}(\text{Conv1D}_{k=5, p=2}(\mathcal{X}_t)))$	$\triangleright H_1 \in \mathbb{R}^{B \times d_{\text{model}} \times L_x}$
4: $H_2 \leftarrow \text{GELU}(\text{IN}(\text{DW-Conv1D}_{k=3, p=1}(H_1)))$	$\triangleright H_2 \in \mathbb{R}^{B \times d_{\text{model}} \times L_x}$
5: $Z \leftarrow R + H_2$	$\triangleright Z \in \mathbb{R}^{B \times d_{\text{model}} \times L_x}$
6: $\mathcal{X}'_t \leftarrow \text{Permute}^{-1}(Z)$	$\triangleright \mathcal{X}'_t \in \mathbb{R}^{B \times L_x \times d_{\text{model}}}$
7: return \mathcal{X}'_t	

Комментарий. Здесь IN обозначает слой InstanceNorm1d [12] с обучаемыми параметрами (affine = True), а DW-Conv1D - глубинную одномерную свёртку по временной оси (groups = d_{model}).