УДК 519.25, 519.7

*А. Ю. Бишук, О. Ю. Бахтеев*

Московский физико-технический институт (национальный исследовательский университет)

# Выбор оптимальной структуры автокодировщика с применением методой байесовской оптимизации

В работе рассматривается задача выбора структуры модели автокодировщика. Под автокодировщиком понимается дифференцируемая по параметрам модель, представимая в виде композиции двух функций: кодировщика, представляющего входной объект в виде скрытого векторного представления, и декодировщика, преобразующего скрытое векторное представление в исходное признаковое пространство. Структура модели автокодировщика представляется в виде набора гиперпараметров, для выбора которых предлагается применять методы байесовской оптимизации. Предлагается двухэтапная модификация метода байесовской оптимизации: на каждой итерации поиска выбирается множество точек с наилучшей оценкой качества модели, а затем из них отбирается наилучший с учетом динамики обучения. Приводится теоретическое обоснование алгоритма, а эксперименты на выборках CIFAR и Fashion-MNIST подтверждают эффективность предложенного подхода.

**Ключевые слова:** байесовская оптимизация, оптимизация гиперпараметров, нейронные сети, AutoML, KDE, TPE, автокодировщик, выбор структуры модели

*A. Y. Bishuk, O. Y. Bakhteev*

Moscow Institute of Physics and Technology

# Optimal Autoencoder Structure Selection Using Bayesian Optimization

The paper considers the problem of the structure selection for autoencoder model. An autoencoder is a model differentiable by parameters, represented as a composition of two functions: an encoder representing the input object as a latent vector representation, and a decoder transforming the latent vector representation into the original feature space. The structure of the autoencoder model is represented as a set of hyperparameters, for the selection of which it is proposed to apply Bayesian optimization methods. A two-stage modification of the Bayesian optimization method is proposed: at each search iteration, a set of points with the best estimate of the model quality is selected, and then the best one is selected from them taking into account the dynamics of training. The theoretical justification of the algorithm is given, and experiments on CIFAR and Fashion-MNIST samples confirm the effectiveness of the proposed approach.

**Key words:** bayesian optimization, hyperparameter optimization, neural networks, AutoML, KDE, TPE, autoencoder, model structure selection

## 1. Введение

Оптимизация архитектуры и гиперпараметров автокодировщиков остается важной и сложной задачей в области машинного обучения. Под автокодировщиком понимается нейронная сеть, состоящая из двух частей — кодировщика и декодировщика, которая используется для обучения эффективных представлений данных без учителя. В фундаментальном

исследовании трудностей обучения глубоких сетей [1] было показано, что выбор гиперпараметров существенно влияет на качество работы модели. Для автокодировщиков эта проблема особенно актуальна, поскольку качество получившихся сжатых представлений сильно зависит от выбранной структуры сети, характеризующейся количеством слоев и их параметрами, размером скрытого представления и другими.

Ранее было предложено несколько специализированных методов оптимизации гиперпараметров глубоких сетей. Например, был разработан алгоритм поиска архитектуры на основе мета-обучения [2], а также для этой задачи были адаптированы [25] древовидные структуры Парзена и разработан подход [4], использующий экстраполяцию кривых обучения для предсказания конечного качества модели. Однако эти методы либо требуют значительных вычислительных ресурсов, либо не учитывают важные особенности динамики обучения конкретных моделей, например, автокодировщиков.

Классические методы оптимизации гиперпараметров, такие как полный перебор [5] и случайный поиск [16], демонстрируют ограниченную эффективность при работе с большими пространствами гиперпараметров. Более перспективным направлением стала байесовская оптимизация [7], которая строит вероятностную модель целевой функции. Развитие этого подхода привело к созданию различных модификаций, включая гауссовские процессы [8] и методы на основе древовидных структур [25].

Несмотря на значительный прогресс в области байесовской оптимизации и нейроархитектурного поиска (NAS) [11], существующие методы имеют несколько принципиальных ограничений применительно к задаче настройки автокодировщиков. Во-первых, большинство подходов, включая современные методы NAS [12], разрабатывались преимущественно для однонаправленных архитектур и не учитывают специфику двунаправленной структуры автокодировщиков [13]. Во-вторых, традиционные методы требуют полного обучения модели для точной оценки качества, что вычислительно дорого для глубоких автокодировщиков. В-третьих, они не учитывают динамику изменения функции потерь на ранних этапах обучения, которая содержит важную информацию о потенциальном качестве модели [14]. Следует отметить, что существуют альтернативные подходы к оптимизации архитектур автокодировщиков, такие как эволюционные алгоритмы [23], однако они требуют значительных вычислительных ресурсов и не обеспечивают теоретических гарантий сходимости.

В нашей работе предлагается новый метод байесовской оптимизации, специально разработанный для настройки гиперпараметров автокодировщиков. Данный подход сочетает двухэтапный отбор кандидатов с анализом динамики обучения на ранних итерациях. На первом этапе отбирается множество перспективных наборов гиперпараметров, на втором — производится их уточнение на основе признаков, извлекаемых после одной эпохи обучения.

Эксперименты на стандартных наборах данных CIFAR [9] и Fashion-MNIST [10] [1] показывают, что предложенный метод превосходит по эффективности базовые подходы байесовской оптимизации для автоэнкодера. В частности, предлагаемый алгоритм демонстрирует более быструю сходимость и позволяет экономить до 60% вычислительных ресурсов при сохранении качества результатов. Эти преимущества особенно важны для практических применений, где требуется частая настройка гиперпараметров автокодировщиков в условиях ограниченных ресурсов. Основной вклад работы заключается в разработке специализированного алгоритма оптимизации гиперпараметров автокодировщиков, его теоретического обоснования, а также экспериментального подтверждения эффективности на стандартных наборах данных.

## 2. Постановка задачи

В данной работе рассматривается задача выбора гиперпараметров модели автокодировщика, задающих его структуру. Пусть задана выборка $\mathbf{X} \subset \mathbb{R}^n$, где $\mathbb{R}^n$ — признако-

[1] Исходный код экспериментов доступен в репозитории https://github.com/intsystems/BHPO-AE

вое пространство. Выборка разделена на обучающую часть $\mathbf{X}_{\text{train}}$ и валидационную часть $\mathbf{X}_{\text{valid}}$. Обучающая часть выборки $\mathbf{X}_{\text{train}}$ используется для оптимизации параметров автокодировщика, а валидационная часть $\mathbf{X}_{\text{valid}}$, размера $M$, для выбора гиперпараметров. Под автокодировщиком понимается параметризованная модель $f_{\mathbf{h}}$, заданная композицией двух отображений:

$$f_{\mathbf{h}}(\mathbf{w}) = \mathbf{f}^{\text{d}}_{\mathbf{h}^d}(\mathbf{w}^{\text{d}}) \circ \mathbf{f}^{\text{e}}_{\mathbf{h}}(\mathbf{w}^{\text{e}}) : \mathbb{R}^n \to \mathbb{R}^n,$$

где $\mathbf{f}^{\text{e}} : \mathbb{R}^n \to \mathbb{R}^m$ — кодировщик с параметрами $\mathbf{w}^e$ и структурой, задаваемой вектором гиперпараметров $\mathbf{h}_e$, $\mathbf{f}^{\text{d}} : \mathbb{R}^m \to \mathbb{R}^n$ — декодировщик с параметрами $\mathbf{w}^d$ и структурой, задаваемой вектором гиперпараметров $\mathbf{h}_d$, $\mathbb{R}^m$ — скрытое пространство ($m \ll n$). Оптимизационная задача для базовой модели автокодировщика выглядит следующим образом:

$$L(\mathbf{h}) = \sum_{\mathbf{x} \in \mathbf{X}_{\text{train}}} ||\mathbf{f}_{\mathbf{h}}(\mathbf{x}) - \mathbf{x}||_2^2 \to \min_{\mathbf{w} \in \mathbb{R}^u},$$

где $\mathbf{w}$ — вектор параметров автокодировщика, образованный конкатенацией векторов $\mathbf{w}^e$ и $\mathbf{w}^d$, $\mathbf{h}$ — вектор гиперпараметров, образованный конкатенацией векторов $\mathbf{h}^e$ и $\mathbf{h}^d$. Заметим, что к модификациям базовой модели автокодировщика относятся также порождающие модели различных классов, включая вариационные автокодировщики [25] и диффузионные модели [26].

Задача оптимизации гиперпараметров формулируется как задача выбора вектора гиперпараметров $\mathbf{h}^* \in \mathbb{H}$, минимизирующего целевую функцию $L(\mathbf{h})$ для валидационной выборки $\mathbf{X}_{\text{valid}}$, характеризующую качество работы модели на данных. Здесь $\mathbb{H}$ представляет пространство допустимых гиперпараметров, которое может включать как непрерывные, например, скорость обучения, так и дискретные параметры, например, количество слоёв.

Ключевая сложность задачи заключается в том, что функция $L(\mathbf{h})$ не имеет аналитического выражения, а её оценка требует дорогостоящей процедуры обучения модели. Кроме того, $L(\mathbf{h})$ в общем случае является невыпуклой, многоэкстремальной и может содержать шум, связанный со стохастичностью обучения. Эти свойства исключают применение базовых методов оптимизации и требуют специальных подходов.

## 2.1. Байесовский выбор гиперпараметров

В рамках байесовского подхода задача оптимизации гиперпараметров модели формулируется как задача нахождения апостериорного распределения:

$$p(\mathbf{h}|\mathbf{X}_{\text{valid}}) = \frac{p(\mathbf{X}_{\text{valid}}|\mathbf{h})p(\mathbf{h})}{p(\mathbf{X}_{\text{valid}})},$$

где $p(\mathbf{h})$ — априорное распределение, $p(\mathbf{X}_{\text{valid}}|\mathbf{h})$ — правдоподобие данных, $p(\mathbf{X}_{\text{valid}})$ — маргинальное правдоподобие.

Оптимальные гиперпараметры находятся как:

$$\mathbf{h}^* = \text{argmax}_{\mathbf{h} \in \mathbb{H}} p(\mathbf{h}|\mathbf{X}_{\text{valid}}).$$

На практике прямое вычисление апостериорного распределения затруднительно, поэтому используются различные методы аппроксимации, включая методы Монте-Карло [15] и суррогатные модели [16]. Для оценки плотности многомерной величины $p(\mathbf{h}|\mathbf{X}_{\text{valid}})$ применяется ядерная оценка плотности KDE [25], которая позволяет аппроксимировать распределение $p(\mathbf{h}|\mathbf{X}_{\text{valid}})$ без строгих предположений о его форме. Для выборки $\{\mathbf{h_i}\}_{i=1}^n$ оценка плотности имеет вид

$$\hat{p}(\mathbf{h}) = \frac{1}{nw^h} \sum_{i=1}^{n} \mathbf{K}\left(\frac{\mathbf{h} - \mathbf{h}_i}{w}\right),$$

где $\mathbf{K} : \mathbb{R}^h \to \mathbb{R}^+$ — ядерная функция (обычно гауссова), $w > 0$ — ширина окна, $h$ — размерность пространства гиперпараметров. KDE находит применение как для построения априорных распределений, так и для анализа пространства гиперпараметров [17].

Основные преимущества KDE в контексте байесовской оптимизации включают непараметрический характер оценки и способность адаптироваться к сложным многомодальным распределениям, что особенно важно при работе с нерегулярными пространствами гиперпараметров.

### 2.2. Метод Tree-structured Parzen Estimator (TPE)

В контексте байесовской оптимизации с использованием KDE возникает ключевая проблема: традиционные методы оценивают единое распределение $p(\mathbf{h}|\mathbf{X}_{\text{valid}})$, что может быть неэффективно для многоэкстремальных функций. Метод Tree-structured Parzen Estimator (TPE) [25] формулирует задачу оптимизации гиперпараметров в терминах условных вероятностей. В отличие от стандартного байесовского подхода, TPE строит раздельные распределения для векторов гиперпараметров с низким значением функции потерь и с высоким. Пусть $\gamma$ — квантиль, разделяющий наблюдения на две группы: $S^+ = \{\mathbf{h}_i | L(\mathbf{h}_i) \leq L^\gamma\}$ и $S^- = \{\mathbf{h}_i | L(\mathbf{h}_i) > L^\gamma\}$. Алгоритм моделирует распределения:

$$p(\mathbf{h}|L) = \begin{cases} p^+(\mathbf{h}) = \frac{1}{|S^+|} \sum_{\mathbf{h}_i \in S^+} \mathbf{K}(\mathbf{h}, \mathbf{h}_i) & \text{если } L(\mathbf{h}) \leq L^\gamma, \\ p^-(\mathbf{h}) = \frac{1}{|S^-|} \sum_{\mathbf{h}_i \in S^-} \mathbf{K}(\mathbf{h}, \mathbf{h}_i) & \text{иначе,} \end{cases}$$

где $\mathbf{K}$ — ядерная функция. Оптимальная следующая точка выбирается по критерию Expected Improvement:

$$\mathbf{h}_{new} = \text{argmax}_{\mathbf{h}} EI(\mathbf{h}) = \text{argmax}_{\mathbf{h}} \frac{p^+(\mathbf{h})}{p^-(\mathbf{h})}. \tag{1}$$

Это позволяет эффективно балансировать между исследованием новых областей и уточнением уже найденных перспективных наборов гиперпараметров.

## 3. Предлагаемый метод

В данном разделе представлен новый алгоритм байесовского выбора гиперпараметров автокодировщиков, сочетающий идеи стохастической оптимизации и слабоконтролируемого обучения [18]. Метод основан на двухэтапной процедуре, где сначала отбираются кандидаты для старта обучения, а затем по первым эпохам определяется лучший вариант для полного обучения. Основной вклад заключается в использовании промежуточных показателей после короткого обучения моделей-кандидатов, что позволяет сократить общие вычислительные затраты по сравнению с базовым подходом.

### 3.1. Описание метода

Предлагаемый метод решает ключевую проблему традиционных подходов к подбору гиперпараметров — необходимость полного обучения множества моделей-кандидатов перед принятием решения об их качестве. Вместо этого мы предлагаем адаптивную стратегию, где решение о перспективности тех или иных гиперпараметров принимается на основе их поведения после ограниченного числа шагов обучения.

Основная гипотеза, лежащая в основе метода, заключается в том, что динамика изменения функции потерь на ранних этапах обучения содержит достаточно информации для прогнозирования конечного качества модели. Это позволяет существенно сократить вычислительные затраты, избегая полного обучения заведомо неперспективных наборов гиперпараметров.

Пусть задано пространство гиперпараметров $\mathbb{H} \subset \mathbb{R}^h$ и функция потерь $L(\mathbf{h}, T)$, оценивающая качество модели после $T$ эпох. Для каждого кандидата $\mathbf{h}$ вычисляются динамические признаки $\hat{\mathbf{u}} = \mathbf{u}(\mathbf{h}, k) \in \mathbb{R}^s$ после $k \ll T$ эпох обучения. Метод решает задачу:

$$\mathbf{h}^* = \text{argmin}_{\mathbf{h}\in\mathbb{H}} E[L(\mathbf{h}, T)]$$

с ограничениями на вычислительный бюджет и частичное обучение (Алгоритм 1). Ключевая гипотеза заключается в том, что динамика изменения $L$ на ранних этапах содержит информацию для прогнозирования конечного качества, что позволяет избегать полного обучения моделей с бесперспективными наборами гиперпараметров.

**Алгоритм 1** Двухэтапный выбор гиперпараметров с раширенным признаковым пространством

**Require:**
$\mathbb{H}$ — пространство гиперпараметров
$l$ — число кандидатов на итерацию
$n$ — число эпох частичного обучения
$L$ — функция потерь
$\mathbf{u}$ — функция вычисления признаков
**Ensure:**
$\mathbf{h}^*$ — оптимальные гиперпараметры
**Инициализация:**
Генерировать начальную популяцию $\{\mathbf{h}_i\}_{i=1}^N \sim p_0(\mathbf{h})$
**for** $i = 1 \dots N$ **do**
    Обучать модель с $\mathbf{h}_i$ в течение $n$ эпох, получить $L_i$
    Вычислить признаки $\hat{\mathbf{u}}_\mathbf{i} = \mathbf{u}(\mathbf{h}_i, L_i)$
**end for**
Построить TPE $p(\mathbf{h}, \hat{\mathbf{u}}|L)$
**Основной цикл:**
**while** критерий останова не достигнут **do**
    **Отбор кандидатов:**
    Выбрать $l$ точек $\{\mathbf{h}_j\}_{j=1}^l$ из $\text{argmax}_\mathbf{h} p(\mathbf{h}, \hat{\mathbf{u}}|L)$
    **Частичное обучение:**
    **for** $j = 1 \dots l$ **do**
        Обучить модель $\mathbf{h}_j$ на $n$ эпох
        Обновить $L_j$, вычислить $\hat{\mathbf{u}}_\mathbf{j}$
    **end for**
    **Обновление модели:**
    Пересчитать $p(\mathbf{h}, \hat{\mathbf{u}}|L)$ на основе расширенного пространства признаков
    Выбрать $\mathbf{h}^* = \text{argmin}_\mathbf{h} \sum_X [L|\mathbf{h}, \hat{\mathbf{u}}]$
**end while**
**Возврат:** $\mathbf{h}^*$

Таким образом, предложенный алгоритм подбора гиперпараметров, легко встраивается в существующие методы путем повторного вызова метода ранжирования объектов, например, TPE. Это делает его гибким с точки зрения использования.

### 3.2. Анализ предложенного метода

Рассмотрим задачу байесовской оптимизации гиперпараметров модели, где целью является нахождение значений $\mathbf{h} \in \mathbb{H}$, минимизирующей итоговую функцию потерь $L(\mathbf{h})$. Предположим, что к каждому набору гиперпараметров $\mathbf{h}$ могут быть сопоставлены два типа признаков: $\mathbf{z}_{\text{naive}}(\mathbf{h})$ — базовые признаки (например, сами гиперпараметры) и

$\mathbf{z}_{\text{hybrid}}(\mathbf{h}) = (\mathbf{z}_{\text{naive}}(\mathbf{h}), \mathbf{g_h})$ — расширенные признаки, включающие информацию $\mathbf{g_h}$, полученную после частичного обучения модели на $k$ эпох (например, промежуточное значение функции потерь, темп сходимости и др.).

Пусть на базе этих признаков строятся ядерные оценки плотности $p_{\text{up}}(\mathbf{h})$ и $p_{\text{down}}(\mathbf{h})$ — оценки плотности элементов с низким (top-25%) и высоким (bottom-75%) значением функции потерь соответственно, а за функцию ранжирования примем EI (1).

Обозначим:

$$R_{\text{naive}}(\mathbf{h}) = \frac{p_{\text{up}}^{\text{naive}}(\mathbf{h})}{p_{\text{down}}^{\text{naive}}(\mathbf{h})}, \quad R_{\text{hybrid}}(\mathbf{h}) = \frac{p_{\text{up}}^{\text{hybrid}}(\mathbf{h})}{p_{\text{down}}^{\text{hybrid}}(\mathbf{h})},$$

где плотности вычислены по соответствующим признакам.

**Теорема** (об асимптотическом неухудшении ранжирования при расширении статистики)

*Пусть для* $\mathbf{h} \in \mathbb{H}$ *даны* $T_1 = \mathbf{z}_{\text{naive}}(\mathbf{h})$ *— базовые признаки и* $T_2 = (\mathbf{z}_{\text{naive}}(\mathbf{h}), \mathbf{g_h})$ *— расширенные признаки, где* $\mathbf{g_h}$ *— результат частичного обучения.*

*Разобьём множество моделей на два класса*:

$$U = \{L(\mathbf{h}) \text{ в top-25\%}\}, \quad B = \{L(\mathbf{h}) \text{ в bottom-75\%}\}.$$

*Определим для* $i \in \{\text{naive}, \text{hybrid}\}$

$$R_i(\mathbf{h}) = \frac{p_{\text{up}}^{(i)}(\mathbf{h})}{p_{\text{down}}^{(i)}(\mathbf{h})},$$

*где плотности* $p_{\text{up}}^{(i)}$ *и* $p_{\text{down}}^{(i)}$ *оцениваются по* $T_i$ *с помощью KDE.*

*Если* $M \to \infty$ *и выполняются стандартные условия консистентности KDE (*$M \to \infty$ *и стандартных условиях [28] на гладкость,* $h_M \to 0$ *и* $Mh_M^d \to \infty$*), то для любых* $\mathbf{h}_1, \mathbf{h}_2 \in \mathbb{H}$ *с* $L(\mathbf{h}_1) < L(\mathbf{h}_2)$:

$$p\big(R_{\text{hybrid}}(\mathbf{h}_1) > R_{\text{hybrid}}(\mathbf{h}_2) \,\big|\, L(\mathbf{h}_1) < L(\mathbf{h}_2)\big) \;\geqslant\; p\big(R_{\text{naive}}(\mathbf{h}_1) > R_{\text{naive}}(\mathbf{h}_2) \,\big|\, L(\mathbf{h}_1) < L(\mathbf{h}_2)\big).$$

**Доказательство:**

При выполнении стандартных условий консистентности KDE:

$$\hat{p}_{\text{up}}^{(i)}(t) \xrightarrow{\text{п.н.}} p(t \mid U), \quad \hat{p}_{\text{down}}^{(i)}(t) \xrightarrow{\text{п.н.}} p(t \mid B).$$

Следовательно, $\hat{R}_i \to R_i^*$.

По построению $T_1$ проекция $T_2$: $T_1 = \pi(T_2)$, поэтому из неравенства обработки информации [27]:

$$I(L; T_2) \geqslant I(L; T_1).$$

Если записать взаимную информацию через энтропии:

$$I(L; T_i) = H(L) - H(L \mid T_i) \quad \Rightarrow \quad H(L \mid T_2) \leqslant H(L \mid T_1).$$

Значит, $T_2$ даёт не меньшую апостериорную определённость о $L$, чем $T_1$.

Рассмотрим задачу попарной классификации: для фиксированной пары $\mathbf{h}_1, \mathbf{h}_2$ определим бинарную метку:

$$Y = \begin{cases} 1, & L(\mathbf{h}_1) < L(\mathbf{h}_2), \\ 0, & \text{иначе.} \end{cases}$$

Байесовский классификатор по $T_i$ минимизирует риск $p(\hat{Y} \neq Y)$ и основан на тесте отношения правдоподобий $\frac{p(Y=1|T_i)}{p(Y=0|T_i)}$.

Так как $T_2$ информативнее ($H(L \mid T_2) \leqslant H(L \mid T_1)$), минимальный риск при $T_2$ не больше, чем при $T_1$ [27]. Значит, вероятность правильного решения $p(\hat{Y} = Y)$ при $T_2$ не меньше.

Наконец, при $M \to \infty$ KDE-оценки сходятся к истинным плотностям. Поэтому процедура, использующая $\hat{R}_i$, реализует оптимальный байесовский классификатор на $T_i$. Следовательно, в пределе вероятность корректного ранжирования, реализуемая практической процедурой на основе $T_2$ (то есть $R_{\text{hybrid}}$), не меньше, чем у процедуры на основе $T_1$ (то есть $R_{\text{naive}}$). Это формально даёт требуемое неравенство:

$$p\big(R_{\text{hybrid}}(\mathbf{h}_1) > R_{\text{hybrid}}(\mathbf{h}_2) \mid L(\mathbf{h}_1) < L(\mathbf{h}_2)\big) \;\geqslant\; p\big(R_{\text{naive}}(\mathbf{h}_1) > R_{\text{naive}}(\mathbf{h}_2) \mid L(\mathbf{h}_1) < L(\mathbf{h}_2)\big).$$

**Замечание (о чувствительности к выбору признаков):** Эффективность метода существенно зависит от информативности используемых признаков $\hat{\mathbf{u}}$. На практике рекомендуется включать в $\hat{\mathbf{u}}$ не только значения функции потерь, но и производные показатели, например, скорость уменьшения функции потерь, совместную информацию между входом и выходом и т.д.

## 4. Вычислительный эксперимент

В данном разделе представлены результаты экспериментальной оценки предложенного алгоритма подбора гиперпараметров нейронных сетей – вариационного и обычного автокодировщиков. Эксперименты проводились на стандартных наборах данных CIFAR и Fashion-MNIST и заключались в сравнении предложенного метода с базовым алгоритмом TPE.

### 4.1. Наборы данных

Для проверки эффективности предложенного метода использовалось два набора данных CIFAR-10 [9] и Fashion-MNIST [10]. CIFAR-10 — набор данных для задач классификации изображений, содержащий 60 000 цветных изображений размером $32 \times 32$ пикселей, разделённых на 10 классов. Набор включает 50 000 обучающих и 10 000 тестовых изображений. Fashion-MNIST — набор данных, содержащий изображения одежды и аксессуаров размером $28 \times 28$ пикселей в градациях серого. Набор включает 60 000 обучающих и 10 000 тестовых примеров, разделённых на 10 классов.

Оба набора данных широко используются для оценки алгоритмов машинного обучения, что позволяет сравнивать предложенный метод с существующими подходами.

### 4.2. Параметры автокодировщика и дополнительные признаки

В процессе оптимизации рассматривались следующие гиперпараметры: количество слоев в кодировщике и декодировщике, размерность скрытого пространства, характеристики свёрточных слоев, включая размеры отступов, размеры ядер и шагов свёртки , а также статистические показатели количества фильтров в свёрточных слоях — максимальное, минимальное и среднее значения, наряду со средним изменением. Дополнительно учитывались общее число весов модели и степень сжатия данных.

Для формирования расширенных признаков $\mathbf{z}_{\text{hybrid}}(x)$, извлекаемых в ходе начальных этапов обучения, использовались различные признаки динамики обучения: значения функции потерь, а для вариационных автокодировщиков дополнительно KL-дивергенция. Также вычислялись показатели совместной информации и канонической корреляции между входными данными, выходными данными и скрытыми представлениями. Дополнительно анализировались квантильные распределения весовых коэффициентов кодировщика и декодировщика вместе с коэффициентами эксцесса для этих распределений.

**Замечание:** Для минимизации затрат на вычисления дополнительных признаков можно использовать только функции потерь на $k$-ой итерации и динамику их изменения, если $k > 1$.

### 4.3. Протокол эксперимента

Эксперимент проводился следующим образом:

**Инициализация:** На каждой итерации выбиралась случайная подвыборка из 5 элементов обучающего множества и обучалась на 100 эпохах. Рассчитывалось среднее значение функции потерь на получившихся элементах.

**Обучение TPE:** По этой подвыборке вычислялись целевые значения для обучения ядерной оценки плотности (TPE). **Отбор кандидатов:**

- **Классический метод:** выбирались кандидаты на основе исходного ранжирования TPE.
- **Предложенный метод:** изначально отбиралось $k$ кандидатов при помощи ранжированием базовым TPE, затем эти кандидаты дообучались на одной эпохе. После чего на основе их функций потерь и дополнительных признаков проводился финальный отбор через ранжирование TPE.

**Обновление обучающего набора данных:** Выбранный кандидат обучался на 100 эпохах и добавлялся в множество для обучения. После чего перерассчитывалось среднее значение функции потерь на размеченных элементах. **Критерий качества:** Динамика средней функции потерь отслеживалась в течение $N$ итераций эксперимента. Кроме того, по окончании эксперимента рассчитывалась средняя площадь под графиком средней функции потерь для базового алгоритма ($S_{naive}$) и предложенного ($S_{hybrid}$). Разница между площадями от первой до 150 итерации алгоритма, а также число шагов, необходимое для достижения минимального значения функции потерь базового алгоритма, также использовались как критерий качества. **Алгоритмы для сравнения:** В эксперименте сравнивались базовый алгоритм TPE, предложенный метод Proposed($l$), где $l$ — число кандидатов на первом этапе, и метод Full, который заранее имел доступ ко всем базовым и рассчитанным статистикам. Стоит отметить, что алгоритм Full является исключительно теоретическим и не может быть использован на практике, а в работе представлен исключительно для сравнения.

Эксперимент повторялся 256 раз, а результат для каждого номера итерации усреднялся для получения статистической значимости результатов.

### 4.4. Результаты

Для демонстрации эффективности предложенного алгоритма были построены графики (рис. 1) со средним значением функции потерь в зависимости от числа размеченных элементов. Использование среднего значения функции потерь, а не лучшего предсказанного на итерации, вызвано тем, что графики становятся более гладкими и интерпретируемыми. Резкое падение функции потерь вызвано тем, что инициализация первых элементов случайна. Поскольку эксперимент проводился на ограниченном наборе гиперпараметров, то рост среднего значения функции потерь в размеченных данных вызван тем, что после выбора лучших кандидатов алгоритмы выбирали из оставшихся, на которых ошибка была выше.

Результаты эксперимента подтвердили эффективность предложенного метода. На обоих наборах данных среднее значение функции потерь размеченных моделей при использовании предложенного алгоритма оказалось ниже, чем при базовом TPE-подходе.

Как видно из таблиц 1 и 2, предложенный алгоритм демонстрирует статистически значимое улучшение по сравнению с базовым методом. На рисунке 1 графики динамики изменения среднего значения функции потерь на размеченных данных также показывают, что предложенный метод быстрее сходится к меньшим значениям функции потерь.

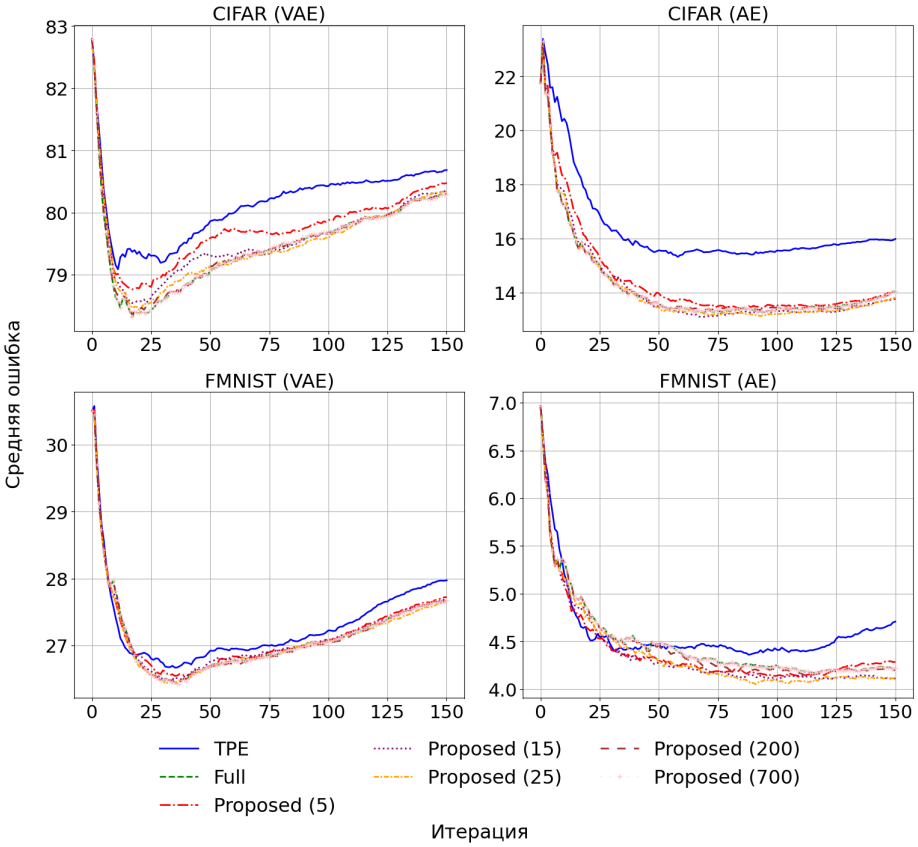


Рис. 1. График среднего значения функции потерь на наборе данных из отобранных алгоритмом и обученных моделей

Таблица 1

**Разница площадей под графиками среднего значения функции потерь размеченной выборки от числа итераций. Рассмотрено на числе итераций, равным 150**

| Набор данных | Модель | $\lvert S_{TPE}\rvert - \lvert S_{Full}\rvert$ | $\lvert S_{TPE}\rvert - \lvert S_{Proposed(5)}\rvert$ | $\lvert S_{TPE}\rvert - \lvert S_{Proposed(15)}\rvert$ |
|---|---|---|---|---|
| Fashion-MNIST | AE | 24.10 ± 8.13 | 31.42 ± 7.58 | 41.25 ± 7.53 |
| Fashion-MNIST | VAE | 29.28 ± 6.91 | 19.16 ± 7.03 | 25.46 ± 7.02 |
| CIFAR-10 | AE | 314.59 ± 21.97 | 277.53 ± 19.35 | 325.72 ± 20.72 |
| CIFAR-10 | VAE | 101.19 ± 10.65 | 50.42 ± 10.88 | 86.38 ± 10.35 |

Таблица 2

**Число шагов для достижения лучшего значения функции потерь базового метода с процентным уменьшением относительно TPE. SBTB — это Steps Best To Baseline, число шагов для достижения лучшего результата базового алгоритма**

| Набор данных | Модель | SBB TPE | SBB Full | SBB Proposed(5) | SBB Proposed(15) |
|---|---|---|---|---|---|
| Fashion-MNIST | AE | 88 | 61 (-30%) | 36 (-59%) | 34 (-61%) |
| Fashion-MNIST | VAE | 35 | 22 (-37%) | 26 (-25%) | 23 (-34%) |
| CIFAR-10 | AE | 58 | 22 (-62%) | 25 (-57%) | 23 (-60%) |
| CIFAR-10 | VAE | 11 | 8 (-27%) | 10 (-9%) | 9 (-18%) |

## 5. Заключение

В данной работе представлен новый алгоритм байесовского подбора гиперпараметров авторегрессионных сетей, основанный на двухэтапной процедуре отбора. Теоретическая

значимость исследования заключается в разработке формальной постановки задачи, доказательстве соответствующей теоремы об условиях эффективности метода, а также введении нового класса алгоритмов, объединяющих байесовскую оптимизацию с ранним прогнозированием качества моделей. С практической точки зрения предложенный метод демонстрирует существенные преимущества, показывая сокращение вычислительных затрат до 60% по сравнению с базовыми подходами в экспериментах на стандартных наборах данных.

Перспективные направления дальнейших исследований включают разработку оптимальных стратегий выбора признаков для различных классов моделей, создание адаптивных схем определения числа эпох частичного обучения, применение метода для архитектурного поиска (NAS) и его интеграцию с методами метаобучения для эффективного переноса знаний между задачами.

Следует отметить, что основное ограничение метода связано с необходимостью тщательного подбора характеристик частичного обучения, таких как число эпох $n$ и состав признаков $\hat{\mathbf{u}}$, для каждой новой задачи. Однако экспериментальные результаты подтверждают, что на практике для настройки этих гиперпараметров достаточно небольшого числа пробных запусков.

## Список литературы


1. *Erhan D.* [*et al.*]. Why does unsupervised pre-training help deep learning? // Proceedings of the thirteenth international conference on artificial intelligence and statistics. 2010. P. 201–208.
2. *Larochelle H.* [*et al.*]. Exploring strategies for training deep neural networks // Journal of Machine Learning Research. 2009. V. 10(1).
3. *Bergstra J.* [*et al.*]. Algorithms for hyper-parameter optimization // Advances in Neural Information Processing Systems. 2011. V. 24.
4. *Domhan T.* [*et al.*]. Speeding up automatic hyperparameter optimization of deep neural networks by extrapolation of learning curves // IJCAI. 2015. V. 15. P. 3460–3468.
5. *Bergstra J., Bengio Y.* Random search for hyper-parameter optimization // The Journal of Machine Learning Research. 2012. V. 13(1). P. 281–305.
6. *Snoek J.* [*et al.*]. Practical bayesian optimization of machine learning algorithms // Advances in Neural Information Processing Systems. 2012. V. 25.
7. *Shahriari B.* [*et al.*]. Unbounded Bayesian optimization via regularization // Artificial Intelligence and Statistics. 2016. P. 1168–1176.
8. *Williams C.K.I., Rasmussen C.E.* Gaussian processes for machine learning. Cambridge, MA : MIT Press, 2006. V. 2(3).
9. *Krizhevsky A.* [*et al.*]. Learning multiple layers of features from tiny images. Toronto, ON, Canada, 2009.
10. *Xiao H., Rasul K., Vollgraf R.* Fashion-MNIST: a novel image dataset for benchmarking machine learning algorithms // arXiv preprint. 2017. arXiv:1708.07747.
11. *Zoph B., Le Q.V.* Neural architecture search with reinforcement learning // arXiv preprint. 2016. arXiv:1611.01578.
12. *Elsken T.* [*et al.*]. Neural architecture search: A survey // Journal of Machine Learning Research. 2019. V. 20(55). P. 1–21.
13. *Kingma D.P.* [*et al.*]. Semi-supervised learning with deep generative models // Advances in Neural Information Processing Systems. 2014. V. 27.
14. *Goodfellow I.* [*et al.*]. Deep learning. Cambridge: MIT Press, 2016. V. 1(2).
15. *Neal R.M.* [*et al.*]. MCMC using Hamiltonian dynamics // Handbook of Markov Chain Monte Carlo. 2011. V. 2(11). P. 2.

**16.** *Snoek J.* [*et al.*]. Practical Bayesian optimization of machine learning algorithms // Advances in Neural Information Processing Systems. 2012. V. 25.

**17.** *Li L.* [*et al.*]. Hyperband: A novel bandit-based approach to hyperparameter optimization // Journal of Machine Learning Research. 2018. V. 18(185). P. 1–52.

**18.** *Zhou Z.-H.* A brief introduction to weakly supervised learning // National Science Review. 2018. V. 5(1). P. 44–53.

**19.** *Silverman B.W.* Density estimation for statistics and data analysis. Routledge, 2018.

**20.** *Domingos P.* A few useful things to know about machine learning // Communications of the ACM. 2012. V. 55(10). P. 78–87.

**21.** *Cover T.M.* Elements of Information Theory. New York : John Wiley & Sons, 1999.

**22.** *Casella G., Berger R.* Statistical Inference. 3rd ed. Boca Raton : Chapman and Hall/CRC, 2024.

**23.** *Suganuma M.* [*et al.*]. A genetic programming approach to designing convolutional neural network architectures // Proceedings of the Genetic and Evolutionary Computation Conference. 2017. P. 497–504.

**24.** *Bergstra J.* [*et al.*]. Algorithms for hyper-parameter optimization // Advances in Neural Information Processing Systems. 2011. V. 24.

**25.** *Bergstra J., Bardenet R., Bengio Y., Kégl B.* Algorithms for hyper-parameter optimization // Advances in Neural Information Processing Systems. 2011. V. 24.

**26.** *Luo C.* Understanding diffusion models: A unified perspective // arXiv preprint. 2022. arXiv:2208.11970.

**27.** *Cover T.M., Thomas J.A.* Entropy, relative entropy and mutual information // Elements of Information Theory. 1991. V. 2(1). P. 12–13.

**28.** *Nakayama K., Silverman G.H.* Serial and parallel processing of visual feature conjunctions // Nature. 1986. V. 320(6059). P. 264–265.

## References

**1.** *Erhan D., et al.*, Why does unsupervised pre-training help deep learning? Proceedings of the thirteenth international conference on artificial intelligence and statistics. 2010. P. 201–208.

**2.** *Larochelle H., et al.*, Exploring strategies for training deep neural networks. Journal of Machine Learning Research. 2009. V. 10(1).

**3.** *Bergstra J., et al.*, Algorithms for hyper-parameter optimization. Advances in Neural Information Processing Systems. 2011. V. 24.

**4.** *Domhan T., et al.*, Speeding up automatic hyperparameter optimization of deep neural networks by extrapolation of learning curves. IJCAI. 2015. V. 15. P. 3460–3468.

**5.** *Bergstra J., Bengio Y.* Random search for hyper-parameter optimization. The Journal of Machine Learning Research. 2012. V. 13(1). P. 281–305.

**6.** *Snoek J., et al.*, Practical bayesian optimization of machine learning algorithms. Advances in Neural Information Processing Systems. 2012. V. 25.

**7.** *Shahriari B., et al.*, Unbounded Bayesian optimization via regularization. Artificial Intelligence and Statistics. 2016. P. 1168–1176.

**8.** *Williams C.K.I., Rasmussen C.E.* Gaussian processes for machine learning. Cambridge, MA : MIT Press, 2006. V. 2(3).

**9.** *Krizhevsky A., et al.*,Learning multiple layers of features from tiny images. Toronto, ON, Canada, 2009.

**10.** *Xiao H., Rasul K., Vollgraf R.* Fashion-MNIST: a novel image dataset for benchmarking machine learning algorithms. arXiv preprint. 2017. arXiv:1708.07747.

**11.** *Zoph B., Le Q.V.* Neural architecture search with reinforcement learning. arXiv preprint. 2016. arXiv:1611.01578.

**12.** *Elsken T., et al.*, Neural architecture search: A survey. Journal of Machine Learning Research. 2019. V. 20(55). P. 1–21.

**13.** *Kingma D.P., et al.*, Semi-supervised learning with deep generative models. Advances in Neural Information Processing Systems. 2014. V. 27.

**14.** *Goodfellow I.* [*et al.*]. Deep learning. Cambridge: MIT Press, 2016. V. 1(2).

**15.** *Neal R.M., et al.*, MCMC using Hamiltonian dynamics. Handbook of Markov Chain Monte Carlo. 2011. V. 2(11). P. 2.

**16.** *Snoek J., et al.*, Practical Bayesian optimization of machine learning algorithms. Advances in Neural Information Processing Systems. 2012. V. 25.

**17.** *Li L., et al.*, Hyperband: A novel bandit-based approach to hyperparameter optimization. Journal of Machine Learning Research. 2018. V. 18(185). P. 1–52.

**18.** *Zhou Z.-H.* A brief introduction to weakly supervised learning. National Science Review. 2018. V. 5(1). P. 44–53.

**19.** *Silverman B.W.* Density estimation for statistics and data analysis. Routledge, 2018.

**20.** *Domingos P.* A few useful things to know about machine learning. Communications of the ACM. 2012. V. 55(10). P. 78–87.

**21.** *Cover T.M.* Elements of Information Theory. New York : John Wiley & Sons, 1999.

**22.** *Casella G., Berger R.* Statistical Inference. 3rd ed. Boca Raton : Chapman and Hall/CRC, 2024.

**23.** *Suganuma M., et al.*, A genetic programming approach to designing convolutional neural network architectures. Proceedings of the Genetic and Evolutionary Computation Conference. 2017. P. 497–504.

**24.** *Bergstra J., et al.*, Algorithms for hyper-parameter optimization. Advances in Neural Information Processing Systems. 2011. V. 24.

**25.** *Bergstra J., Bardenet R., Bengio Y., Kégl B.* Algorithms for hyper-parameter optimization. Advances in Neural Information Processing Systems. 2011. V. 24.

**26.** *Luo C.* Understanding diffusion models: A unified perspective. arXiv preprint. 2022. arXiv:2208.11970.

**27.** *Cover T.M., Thomas J.A.* Entropy, relative entropy and mutual information. Elements of Information Theory. 1991. V. 2(1). P. 12–13.

**28.** *Nakayama K., Silverman G.H.* Serial and parallel processing of visual feature conjunctions. Nature. 1986. V. 320(6059). P. 264–265.

Поступила в редакцию 11.08.2025