Вычислительные технологии, 2025, том 30, № 4, с. 145–158. © ФИЦ ИВТ, 2025
Computational Technologies, 2025, vol. 30, no. 4, pp. 145–158. © FRC ICT, 2025
ISSN 1560-7534
eISSN 2313-691X

ИНФОРМАЦИОННЫЕ ТЕХНОЛОГИИ

DOI:10.25743/ICT.2025.30.4.013

Нейросетевые модели определения сложных для перевода предложений

С. И. Смагин[1], В. Е. Окладников[1], Т. В. Кожевникова[1,*], А. А. Животова[2]
[1]Вычислительный центр ДВО РАН, 680000, Хабаровск, Россия
[2]Комсомольский-на-Амуре государственный университет, 681013, Комсомольск-на-Амуре, Россия
*Контактный автор: Кожевникова Татьяна Владимировна, e-mail: ktvsl@mail.ru
Поступила 21 мая 2025 г., доработана 03 июня 2025 г., принята в печать 11 июня 2025 г.

Рассматривается задача определения плохо переводимых предложений, которая является подзадачей улучшения машинного перевода путем перефразирования оригинала. Ее актуальность обусловлена возрастающей потребностью развития и практического применения систем компьютерного перевода. Исследуются возможности алгоритмов на основе нейронных сетей со слоями эмбеддингов, рекуррентных слоев и слоев одномерной свертки, которые обучены для задачи обнаружения плохо переводимых предложений. Сравниваются архитектуры нейронных сетей, не требующие больших вычислительных затрат, и оценивается целесообразность их использования на практике.

Ключевые слова: нейронная сеть, машинное обучение, алгоритм, машинный перевод, классификация.

Цитирование: Смагин С.И., Окладников В.Е., Кожевникова Т.В., Животова А.А. Нейросетевые модели определения сложных для перевода предложений. Вычислительные технологии. 2025; 30(4):145–158. DOI:10.25743/ICT.2025.30.4.013.

Введение

Современный мир немыслим без машинного перевода, который становится все более востребованным в различных сферах деятельности человека. С ростом объема информации и скорости ее создания и распространения возникает острая необходимость в повышении качества перевода при одновременном сокращении затрат. В настоящее время исследования в области машинного перевода сосредоточены преимущественно на улучшении самих алгоритмов перевода, что подтверждают работы [1, 2]. Однако повышение качества перевода возможно не только за счет усовершенствования моделей перевода, но и путем предварительного предредактирования исходного текста, важной частью которого является выявление “плохо переводимых” предложений [3].

Задача перевода является частным случаем более общей задачи — генерации текста, решаемой языковыми моделями. Современные языковые модели, такие как GPT и BERT, способны не только переводить тексты, но и генерировать связные и осмысленные предложения [4]. Однако, несмотря на высокую эффективность, развертывание, обучение и системное использование таких моделей требуют значительных вычислительных ресурсов. Высокая стоимость обслуживания делает их массовое применение в задачах перевода затруднительными, особенно для небольших компаний с ограниченными ресурсами.

1. Постановка задачи

Предредактирование позволяет оптимизировать текст, делая его более пригодным для обработки машинным переводом. Один из ключевых этапов этого процесса — классификация предложений исходного текста на “хорошо” и “плохо” переводимые. Такая классификация помогает оптимизировать нагрузку на редакторов и переводчиков, снижая объем предложений, требующих ручной корректировки, что особенно важно для малых переводческих компаний с ограниченными ресурсами.

В данной работе рассматриваются архитектуры нейронных сетей, основанные на слоях эмбеддингов, рекуррентных слоях (SimpleRNN, LSTM) и одномерных сверточных слоях (Conv1D). В качестве входных данных используются предложения русского языка, представленные в виде последовательностей токенов. Приводится описание архитектур и слоев нейронных сетей. Точность алгоритмов оценивается с помощью стандартных метрик машинного обучения. Предлагаемый подход упрощает задачу перефразирования и улучшения машинного перевода текста путем снижения числа предложений, требующих специальной обработки.

Выбор рассматриваемых архитектур обусловлен их небольшими эксплуатационными затратами по сравнению с современными трансформерными моделями, такими как BERT, которые требуют значительных вычислительных ресурсов при обучении и точной настройки (fine-tuning). Для малых переводческих компаний использование таких моделей может оказаться затруднительным из-за высокой стоимости развертывания и обслуживания [5].

Цель данного исследования заключается в сравнении архитектур нейронных сетей, не требующих больших вычислительных затрат для задачи определения плохо переводимых предложений, и оценке целесообразности их использования на практике.

2. Метрики качества

Оценка качества моделей проводится с использованием метрики оценки качества машинного перевода hLEPOR (далее — метрика), основанной на n-граммах и учитывающей корреляцию с человеческими оценками перевода. В данном контексте под n-граммами понимается последовательность из n слов в тексте.

Метрика вычисляет сходство n-грамм в машинном переводе и референсном переводе (после редакторских правок) сегмента текста, она изменяется в диапазоне от нуля до 1, где нуль указывает на отсутствие сходства между предложениями, а 1 — на полное сходство предложений. Метрика имеет лучший показатель корреляции Пирсона с человеческими суждениями по языковой паре английский – русский [6].

Метрика вычисляется по следующему алгоритму.

1. Сначала рассчитываются метрики

$$\text{Precision} = \frac{\text{TP}}{\text{TP} + \text{FP}}, \tag{1}$$

$$\text{Recall} = \frac{\text{TP}}{\text{TP} + \text{FN}}, \tag{2}$$

где TP — количество слов (или n-грамм), которые есть и в машинном, и в эталонном переводах; FP — количество слов, которые есть в машинном, но которых нет

в эталонном; FN — количество слов, которые отсутствуют в машинном, но есть в эталонном переводе.

В метриках, аналогичных hLEPOR, величины TP, FP и FN интерпретируются не как бинарные метки классов, а через совпадения n-грамм между машинным переводом (mt) и эталонным переводом (ref).

Метрика Precision (точность) показывает, какая доля слов из машинного перевода есть в эталонном, Recall (полнота) — долю слов из эталонного перевода, присутствующих в машинном.

2. Затем метрики Precision и Recall объединяются через гармоническое среднее в метрику $F1$:

$$F1 = 2\frac{\text{Precision} \cdot \text{Recall}}{\text{Precision} + \text{Recall}} = \frac{\text{TP}}{\text{TP} + 0.5(\text{FP} + \text{FN})}. \quad (3)$$

3. Далее рассчитываются два штрафа: за разницу длины предложений (LP) и за порядок слов (OR):

$$LP = \exp\left(1 - \frac{\max(len_{mt}, len_{ref})}{\min(len_{mt}, len_{ref})}\right), \quad OR = 1 - \frac{1}{n}\sum_{i=1}^{n}\frac{|pos_{mt}(w_i) - pos_{ref}(w_i)|}{\max(len_{mt}, len_{ref})}.$$

Здесь len — функция длины предложения, $pos_{mt}(w_i)$ — позиция слова w_i в машинном переводе, $pos_{ref}(w_i)$ – позиция этого же слова в эталонном переводе, n — число совпадающих слов.

4. Итоговая метрика hLEPOR вычисляется как гармоническое среднее взвешенных штрафов и оценки качества:

$$\text{hLEPOR} = Harmonic(w_{LP} \cdot LP, w_{OR} \cdot OR, w_{F1} \cdot F1) = \frac{3}{\frac{1}{w_{LP} \cdot LP} + \frac{1}{w_{OR} \cdot OR} + \frac{1}{w_{F1} \cdot F1}}.$$

Здесь LP, OR, $F1$ определены выше, а w_{LP}, w_{OR}, w_{F1} — веса параметров для гибкой настройки метрики. В расчетах использовались стандартные веса из источника [7].

3. Исходные данные

Как источник исходных данных используется база данных `Translation Memories` компании-поставщика лингвистических услуг, которая специализируется на переводе технической проектной документации международных нефтегазовых проектов. `Translation Memories` накапливает параллельные двуязычные корпуса в процессе перевода при помощи средств автоматизации переводческой деятельности (`CAT-tools`). Они тщательно проверяются редакторами и повторно используются для обеспечения единообразия переведенной документации и сокращения времени на перевод.

Исходные данные для обучения моделей и тестирования — датасет `Oil_Gas_ru_en` из базы `Translation Memories` объемом 65 420 экземпляров выборки. Это эквивалентно 3271 стандартной странице текста на английском языке (одна страница — 1800 знаков с пробелами). Данные, собранные в датасет, предварительно очищены и анонимизированы. `Oil_Gas_ru_en` включает:

- набор предложений на русском языке (поле `source`);
- набор предложений на английском языке после машинного перевода (поле `target`);

Т а б л и ц а 1. Фрагмент обучающей выборки данных
Table 1. Fragment of the training dataset

№	Предложения			Значения hLEPOR
	source	target	edited	
1	Во избежание заражения не следует касаться обожженных участков кожи руками, вскрывать пузыри, удалять приставшие к обожженному месту вещества	To avoid contamination, the burned areas of the skin should not be touched by hands, bubbles should be opened, and substances attached to the burnt place should be removed	To avoid infection, do not touch the burnt areas or open bubbles, or remove any substances stuck to the burn	0.560
2	Обеспечивает сооружение всех временных (подъездных к участку строительства) дорог и коммуникаций, требуемых для выполнения работ, в соответствии с проектной и/или рабочей документацией, их содержание и ремонт	Provides for the construction, maintenance and repair of all temporary (access roads to the site) roads and communications required for the performance of the work in accordance with the project and/or working documents	Arrange and maintain all temporary (access roads to the site) roads and utilities for the work according to the design and/or working documents	0.587
3	Отклонение от прямолинейности и плоскостности поверхности на длине 1–3 м и местные неровности поверхности бетона	Deviation from straightness and flatness of surface at 1 to 3 m and local surface irregularities of concrete	Offset of surface misalignment and flatness on 1–3 m length and local concrete irregularities	0.623
4	С повышением температуры область воспламенения аммиачно-воздушных смесей расширяется и при 100 °C лежит в интервале 14.5–33.6 %	With an increase in temperature, the ammonium-air mixture ignition area expands to 14.5–33.6 % per volume at 100 °C	Due to temperature rise, the air-ammonia mixture ignition range increases and is 14.5–33.6 % volume at 100 °C	0.488

- набор предложений на английском языке после редактирования (поле `edited`);
- значения метрики, рассчитанные для предложений из набора предложений на английском языке после машинного перевода и предложений из набора предложений на английском языке после редакторских правок (поле hLEPOR).

Фрагмент обучающей выборки приведен в табл. 1.

4. Предобработка

“Плохо переводимыми” предложениями считаются те, которые имеют значение метрики меньше 0.65. При определении “плохо переводимых” предложений можно решать либо задачу регрессии, прогнозируя значение метрики, либо перейти к задаче бинарной классификации, изначально разделив обучающую выборку на “плохо” и “хорошо”

переводимые предложения по уже рассчитанной метрике. В машинном обучении под задачей регрессии понимают задачу, целью которой является построение модели, позволяющей предсказывать значения одной или нескольких выходных переменных на основе значений одной или нескольких входных переменных.

При анализе обучающего набора данных `Oil_Gas_ru_en`, фрагмент которого представлен в табл. 1, выявлено, что число предложений класса "плохо переводимых" превышает количество "хорошо переводимых" и составляет около 60 % выборки. При обучении нейронных сетей необходимо, чтобы эти классы имели сопоставимое количество примеров, иначе модель будет склонна отмечать преобладающий класс — в данном случае "плохо переводимые" предложения. Для устранения этого перекоса избыточные записи класса, имеющего наибольший размер, следует удалить [8]. Нужное число записей удалялось случайным образом, такой подход позволил сохранить репрезентативность выборки, избегая систематического искажения. Дополнительно проверялось влияние балансировки на качество моделей: обучение проводилось как на оригинальном несбалансированном наборе, так и на сбалансированной выборке. Было зафиксировано улучшение значений метрик на сбалансированной выборке, что подтверждает целесообразность применения предварительной балансировки.

Далее выполняется преобразование текста в векторную форму (последовательность токенов). Для этого из библиотеки `tensorflow.keras.preprocessing.text` используют модуль `Tokenizer` с настройками, представленными в табл. 2.

Пример векторного представления предложений оригинального текста на русском языке из колонки source табл. 1 приведен ниже.

1. [76, 1525, 1, 10, 115, 14 432, 1, 482, 3582, 8800, 1, 1, 18 809, 1, 13, 1, 822, 1054].
2. [387, 2539, 63, 714, 6718, 13, 5431, 48, 2628, 2, 1165, 4065, 7, 39, 9, 3, 21, 5, 194, 2, 8, 185, 634, 28, 723, 2, 900].
3. [1757, 12, 18 608, 2, 1, 168, 4, 1422, 14, 10 432, 2, 3028, 16 931, 168, 940].
4. [5, 1, 377, 1898, 6074, 1, 3108, 6276, 1, 2, 11, 245, 9616, 8799, 3, 2973, 544, 30, 1, 58, 12 426].

На следующем шаге исследуется распределение длин векторных представлений предложений по числу токенов. Это необходимо для указания длины входящего в нейронную сеть вектора при создании первого слоя. Максимальная длина последовательности токенов в предложениях из обучающей выборки получилась равной 54. Это максимальное количество слов в одном предложении из обучающей выборки. Для входного слоя нейронной сети принято использовать количество нейронов, кратное степени двойки, поскольку GPU (особенно NVIDIA CUDA) эффективнее обрабатывает векторы длины, кратной степени двойки. Поэтому для стандартизации длины вектора использовалось значение 64 — наиболее близкая к 54 степень двойки с избытком. Все предложения обучающей выборки дополнялись до длины 64. Свободные позиции вектора заполнялись

Т а б л и ц а 2. Настройки модуля Tokenizer
Table 2. Tokenizer settings

Параметр	Значение	Комментарий
`num_words`	20 000	Максимальный размер словаря
`filters`	`!"#$%&()*+,-./:;<=>?@[\]^_`{}|`	Удаляются нежелательные символы
`lower`	True	Приведение текста к нижнему регистру
`split`	" "	Разделитель слов
`oov_token`	`'неизвестное_слово'`	Токен для неизвестных слов

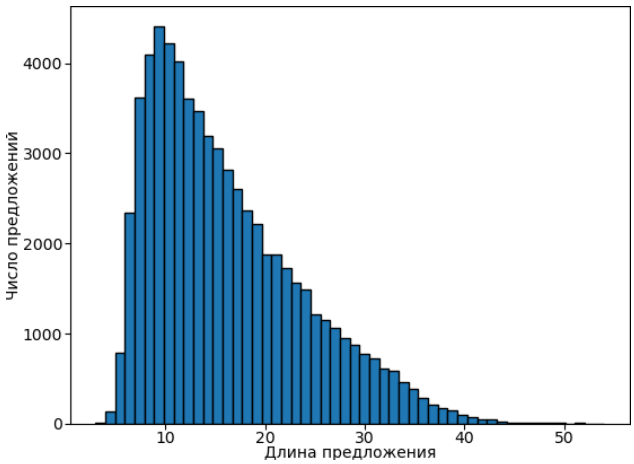


Рис. 1. Распределение предложений обучающей выборки по числу токенов
Fig. 1. Distribution of sentence lengths in the training dataset

нулями (незначащими токенами). Распределение предложений в обучающей выборке по числу токенов приведено на рис. 1.

5. Архитектуры и обучение нейронных сетей

В работе рассматриваются три наиболее известные и часто используемые архитектуры нейронных сетей, использующие слои следующих типов: Embedding, SimpleRNN, Dense, LSTM и Conv1D [9–12]. Для регуляризации используются слои SpatialDropout1D, BatchNormalization и Dropout [13], а для оценки моделей — метрики (1)–(3). Все архитектуры созданы и обучены с помощью библиотеки глубокого машинного обучения Tensorflow-Keras [14].

Первая рассматриваемая архитектура A1 имеет слои: Input, Embedding, SpatialDropout1D, BatchNormalization, SimpleRNN, Dropout, Dense + Softmax. Структура модели отрисована с помощью библиотеки Visualkeras и представлена на рис. 2. Рассмотрим каждый слой этой нейронной сети подробнее.

Input получает входные данные (векторы токенезированных предложений) длиной 64.

Embedding преобразует категориальные векторы (векторы, содержащие дискретные значения вектора, такие как слова, метки или ID) в плотные векторы фиксированной длины (векторы с непрерывными значениями, где большинство элементов ненулевые). Ключевые параметры слоя включают: `input_dim` — размер словаря, т. е. общее

```
Model: "sequential_1"
_________________________________________________________________
Layer (type)                 Output Shape              Param #
=================================================================
embedding_1 (Embedding)      (None, 64, 5)             100000
_________________________________________________________________
spatial_dropout1d_1 (Spatial (None, 64, 5)             0
_________________________________________________________________
batch_normalization_1 (Batch (None, 64, 5)             20
_________________________________________________________________
simple_rnn (SimpleRNN)       (None, 20)                520
_________________________________________________________________
dropout_1 (Dropout)          (None, 20)                0
_________________________________________________________________
dense_1 (Dense)              (None, 2)                 42
=================================================================
Total params: 100,582
Trainable params: 100,572
Non-trainable params: 10
```

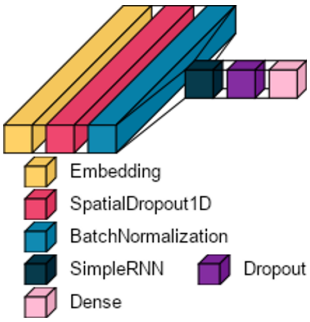


Рис. 2. Архитектура A1 (SimpleRNN)
Fig. 2. Architecture A1 with SimpleRNN

количество уникальных токенов; `output_dim` определяет размер плотного вектора, на который будет отображаться каждый токен; необязательный параметр `input_length` — длина входной последовательности.

SpatialDropout1D — регуляризационный слой, который "выключает" 1D-карты объектов из эмбеддинг-векторов. Так как после Embedding-слоя нельзя использовать слой Dropout, мы используем его альтернативу — SpatialDropout1D. Это вызвано тем, что слой Embedding разворачивает слова в таблицу с тензорами определенной длины. Dropout-слой будет удалять значения из этих тензоров, что нарушит логику работы модели. Слой SpatialDropout1D удаляет весь тензор, также руководствуясь логикой модели.

BatchNormalization нормализует входные данные для каждого батча — подмножества примеров из обучающей части набора данных, которую модель получает во время одной итерации. Это ускоряет обучение нейронной сети и делает ее менее чувствительной к начальной инициализации весов. Данный слой помогает стабилизировать процесс обучения и предотвращает проблемы, связанные с затуханием градиента при обратном распространении ошибки в глубоких нейронных сетях.

SimpleRNN обрабатывает последовательные данные, поочередно принимая элементы входной последовательности и на каждом шаге обновляя свое скрытое состояние h_t, которое зависит как от текущего входа x_t, так и от предыдущего состояния h_{t-1}. Это скрытое состояние выступает как своего рода "память", позволяющая учитывать контекст предыдущих элементов. Благодаря этому SimpleRNN способен улавливать временные зависимости в данных, что делает его полезным для задач, где важны порядок и связь между элементами, такими как анализ текста, временных рядов или аудио.

Dropout — это регуляризационный слой, который во время обучения случайным образом "отключает" (обнуляет) часть нейронов, тем самым предотвращая переобучение и повышая способность модели к обобщению. Это означает, что сеть не может слишком сильно полагаться на конкретные нейроны и учится более устойчивым признакам. Во время инференса (т. е. в случае применения модели к новым данным) все нейроны остаются активными, но их выходы масштабируются (умножаются на коэффициент, соответствующий вероятности сохранения нейронов во время обучения), чтобы сохранить согласованность распределения активаций между обучением и использованием.

Dense — полносвязный слой, выполняющий линейное преобразование входных данных с последующим применением функции активации. Он принимает на вход вектор признаков, умножает его на матрицу весов, добавляет смещение (bias) и полученный вектор передает в функцию активации — математическую функцию, которая определяет, будет ли нейрон в нейронной сети активирован (т. е. передаст сигнал дальше) и каким образом. Функция активации применяется к взвешенной сумме входных данных и смещения (bias) и придает модели способность обучаться сложным зависимостям и нелинейностям. Таким образом, слой Dense извлекает и обобщает важные признаки, трансформируя входные данные в формат, пригодный для следующего слоя или для финального вывода (например, классификации или регрессии).

Softmax — это слой функции активации, который используется на выходе нейросети для задач классификации. Он преобразует выходные значения модели в набор вероятностей, соответствующих каждому классу (например, "хорошо" или "плохо" переведенные предложения), при этом гарантируя, что сумма этих вероятностей равна 1. Такой результат считается более интерпретируемым, потому что он характеризует уровень "уверенности" модели в каждом варианте. Например, можно увидеть, что модель

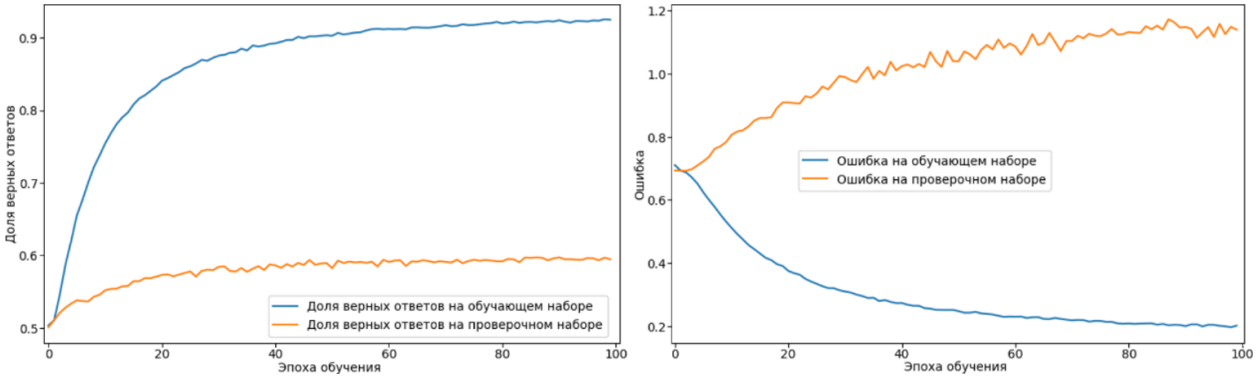
с вероятностью 0.9 считает перевод “хорошим” и с 0.1 — “плохим”. Это помогает лучше понимать и анализировать ее поведение.

После обучения модель A1 показала на тестовой выборке значение метрики $F1 = 0.59$. График обучения и отчет по классификации приведен на рис. 3.

Результаты расчета метрики $F1$ на тестовой выборке для модели A1 принимаются в качестве исходного результата с целью дальнейшего его улучшения и получения значения метрики, близкой к единице.

Обычные рекуррентные нейронные сети испытывают трудности из-за исчезающего градиента ошибки, что приводит к ухудшению обучения на длинных последовательностях.

Перейдем к рассмотрению модели A2, которая получается из модели A1 заменой слоя SimpleRNN на рекуррентный слой LSTM (long short-term memory). Эта замена решает упомянутую выше проблему исчезающего градиента. Слой LSTM включает в себя специальный механизм “запоминания”, который позволяет сети сохранять информацию на протяжении более длительных периодов времени, чем SimpleRNN. Архитектура A2 представлена на рис. 4, график обучения и отчет по классификации приведены на рис. 5.



	Presision	Recall	$F1$	Support
0	0.59	0.60	0.60	8180
1	0.60	0.59	0.59	8175
micro avg	0.59	0.59	0.59	16 355
macro avg	0.59	0.59	0.59	16 355
weighted avg	0.59	0.59	0.59	16 355
samples avg	0.59	0.59	0.59	16 355

Рис. 3. Графики обучения и отчет по классификации архитектуры A1
Fig. 3. Training graphs for architecture A1

```
Model: "sequential_2"
_________________________________________________________________
Layer (type)                 Output Shape              Param #
=================================================================
embedding_2 (Embedding)      (None, 64, 5)             100000
_________________________________________________________________
spatial_dropout1d_2 (Spatial (None, 64, 5)             0
_________________________________________________________________
batch_normalization_2 (Batch (None, 64, 5)             20
_________________________________________________________________
lstm_1 (LSTM)                (None, 20)                2080
_________________________________________________________________
dropout_2 (Dropout)          (None, 20)                0
_________________________________________________________________
dense_2 (Dense)              (None, 2)                 42
=================================================================
Total params: 102,142
Trainable params: 102,132
Non-trainable params: 10
```

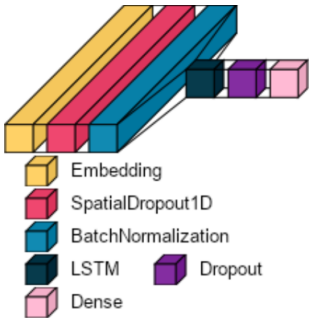


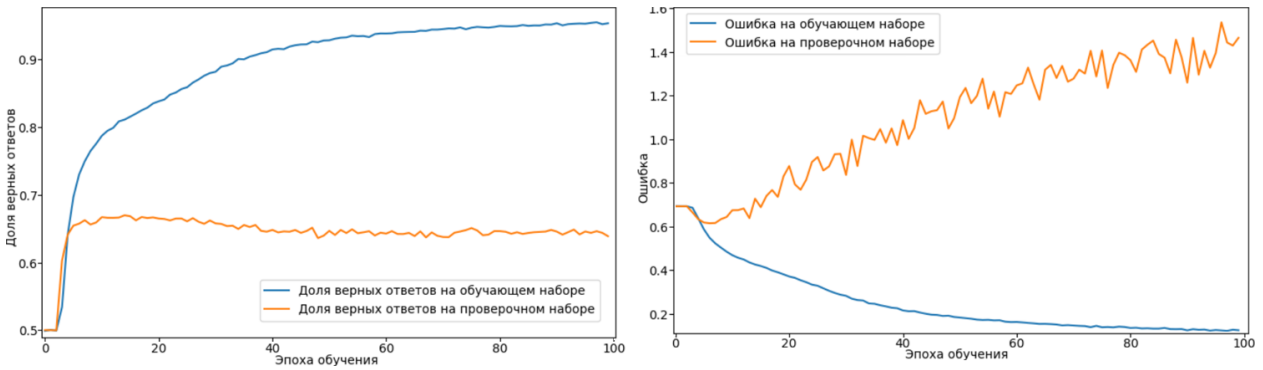
Рис. 4. Архитектура A2 (LSTM)
Fig. 4. Architecture A2 with LSTM

Значение метрики на тестовой выборке для модели A2 составляет 0.64, что на 0.05 больше, чем у A1. Из графиков обучения видно, что модель переобучается и с 20-й эпохи точность на тестовой выборке падает.

Далее, для перехода к следующей архитектуре нейронной сети A3 заменим в A2 слой LSTM на слой одномерной свертки Conv1D.

Одномерная свертка — это операция, которая скользит по последовательности данных, применяя к ней ядро, чтобы извлечь локальные признаки. В задачах NLP (natural language processing) этот слой позволяет нейронным сетям улавливать локальные зависимости между словами в тексте. После сверточного слоя, чтобы передать сигнал в полносвязный слой, необходимо использовать слой Flatten. В нейронных сетях он выполняет простую, но важную функцию: преобразует многомерный входной тензор в вектор.

Архитектура нейронной сети A3 представлена на рис. 6, график обучения и отчет по классификации приведены на рис. 7. Значение метрики $F1$ для A3 составляет 0.68, что на 0.04 больше, чем у A2. Из графиков видно, что модель переобучается и с 10-й эпохи точность на тестовой выборке падает, максимальное значение метрики $F1$ составило 0.68.



	Presision	Recall	$F1$	Support
0	0.66	0.57	0.61	8180
1	0.62	0.71	0.66	8175
micro avg	0.64	0.64	0.64	16 355
macro avg	0.64	0.64	0.64	16 355
weighted avg	0.64	0.64	0.64	16 355
samples avg	0.64	0.64	0.64	16 355

Рис. 5. Графики обучения и отчет по классификации архитектуры A2
Fig. 5. Training graphs for architecture A2

```
Model: "sequential_3"
_________________________________________________________________
Layer (type)                 Output Shape              Param #
=================================================================
embedding_5 (Embedding)      (None, 64, 10)            200000
_________________________________________________________________
spatial_dropout1d_5 (Spatial (None, 64, 10)            0
_________________________________________________________________
batch_normalization_7 (Batch (None, 64, 10)            40
_________________________________________________________________
conv1d_4 (Conv1D)            (None, 64, 20)            1020
_________________________________________________________________
conv1d_5 (Conv1D)            (None, 60, 20)            2020
_________________________________________________________________
max_pooling1d_2 (MaxPooling1 (None, 30, 20)            0
_________________________________________________________________
dropout_3 (Dropout)          (None, 30, 20)            0
_________________________________________________________________
batch_normalization_8 (Batch (None, 30, 20)            80
_________________________________________________________________
flatten_2 (Flatten)          (None, 600)               0
_________________________________________________________________
dense_3 (Dense)              (None, 2)                 1202
=================================================================
Total params: 204,362
Trainable params: 204,302
Non-trainable params: 60
```

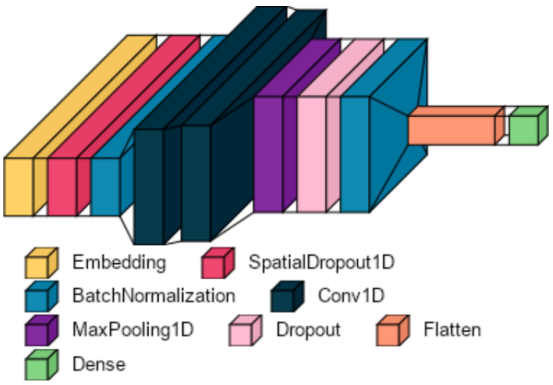
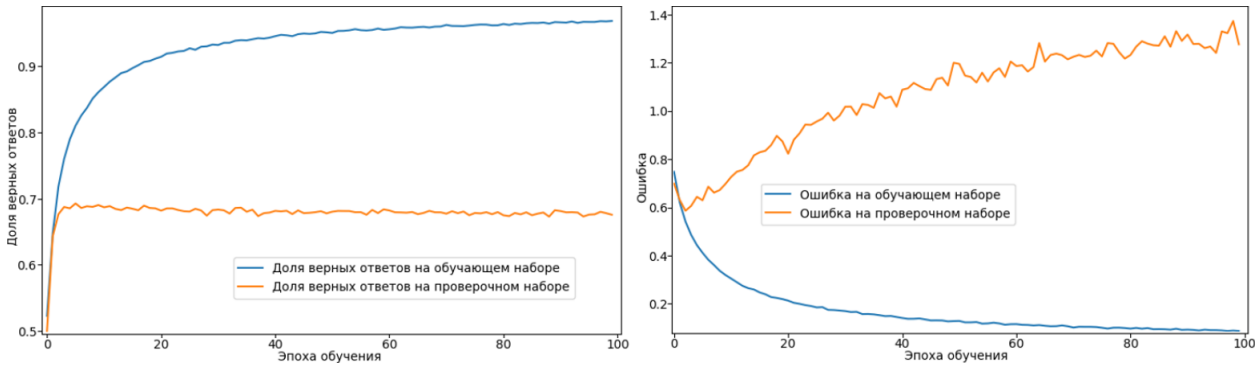


Рис. 6. Архитектура A3 (Conv1D + Flatten)
Fig. 6. Architecture A3 with Conv1D



	Presision	Recall	$F1$	Support
0	0.68	0.68	0.68	8180
1	0.68	0.68	0.68	8175
micro avg	0.68	0.68	0.68	16 355
macro avg	0.68	0.68	0.68	16 355
weighted avg	0.68	0.68	0.68	16 355
samples avg	0.68	0.68	0.68	16 355

Рис. 7. Графики обучения и отчет по классификации архитектуры A3
Fig. 7. Training graphs for architecture A3

Т а б л и ц а 3. Сравнение архитектур по метрике $F1$ и вычислительным затратам
Table 3. Comparison of architectures by $F1$ metric and computational costs

Модель	$F1$	Время обучения, с	Память, МБ (RAM, пик)	Параметры	Размер файла, кБ	Время инференса, с
A1	0.59	352.3	5702	100k	400	0.037
A2	0.64	62	5698	100k	842	0.033
A3	0.68	94	5691	204k	1651	0.029

В табл. 3 приведены значения метрики $F1$ для моделей A1, A2, A3 на тестовой выборке. Так, лучшей архитектурой по показателям метрики является архитектура A3, использующая одномерные свертки.

Небольшие фирмы, занимающиеся переводом, имеют ограниченные вычислительные ресурсы. Именно для их деятельности подходит классификатор, определяющий “плохо” и “хорошо” переводимые предложения. Используемые в этом исследовании архитектуры требуют меньший объем вычислительных ресурсов относительно трансформенных архитектур. Отметим, что в предыдущем исследовании [15], где использовались 68 численных признаков, связанных с общими количественными, морфологическими, синтаксическими и лексическими признаками текста, максимальное значение метрики $F1$ находилось на уровне 0.59. Рассмотренные в данной работе новые модели демонстрируют более высокое качество предсказания класса, достигая значения метрики $F1$, равного 0.69.

Заключение

В работе представлены несколько моделей нейронных сетей для решения проблемы улучшения качества машинного перевода путем перефразирования оригинала. Описаны алгоритмы глубокого машинного обучения на основе моделей рекуррентных и сверточных слоев, которые могут классифицировать предложения русского языка на два класса, а именно на “хорошо”/“плохо” переводимые на английский язык предложения.

Настоящее исследование сосредоточено на анализе архитектур, не требующих значительных вычислительных ресурсов на дообучение, что делает их доступными для применения в малых переводческих компаниях. Значения метрик на тестовой выборке для рассмотренных моделей позволяют рекомендовать разработанные архитектуры к использованию для работы в небольших фирмах, занимающихся переводами. В будущих исследованиях для повышения точности классификации предложений на “плохо” и “хорошо” переводимые предполагается рассмотреть новые и известные предобученные трансформерные модели, такие как BERT.

Список литературы

[1] **Way A.** Quality expectations of machine translation. Moorkens J., Castilho S., Gaspari F., Doherty S. (Eds.) Translation quality assessment. Machine translation: technologies and applications. Cham: Springer International Publishing; 2018; (1):159–178. DOI:10.1007/978-3-319-91241-7_8. Available at: https://link.springer.com/chapter/10.1007/978-3-319-91241-7_8.

[2] **Jiang W.** Pre-editing for machine translation. Proceedings of the 9th Conference of the Association for Machine Translation in the Americas: Government MT User Program. Denver, USA; 2010. Available at: https://aclanthology.org/2010.amta-government.6.

[3] **Животова А.А., Бердоносов В.Д.** Оптимизационное предредактирование узкоспециальных русскоязычных текстов для их машинного перевода на английский язык. Информационно-математические технологии в науке и управлении. 2024; 2(34):169–182. DOI:10.25729/ESI.2024.34.2.016.

[4] **Zaki M.Z.** Revolutionising translation technology: a comparative study of transformer models — BERT, GPT, and T5. Computer Science & Engineering: An International Journal. 2024; 14(3):15–27. DOI:10.5121/cseij.2024.14302. Available at: https://www.cseij.org/papers/v14n3/14324cseij02.pdf.

[5] **Šuppa M., Benešová K., Švec A.** Cost-effective deployment of BERT models in serverless environment. Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies: Industry Papers. 2021: 187–195. DOI:10.18653/v1/2021.naacl-industry.24. Available at: https://aclanthology.org/2021.naacl-industry.24.

[6] **Han A.L.F., Wong D.F., Chao L.S., He L., Lu Y., Xing J., Zeng X.** Language-independent model for machine translation evaluation with reinforced factors. Proceedings of Machine Translation Summit XIV: Posters. Nice, France; 2013. Available at: https://aclanthology.org/2013.mtsummit-posters.3.

[7] **Han L., Sorokina I., Erofeev G., Gladkoff S.** CushLEPOR: customising hLEPOR metric using Optuna for higher agreement with human judgments or pre-trained language model LaBSE. Proceedings of the Sixth Conference on Machine Translation. 2021: 1014–1023. Available at: https://aclanthology.org/2021.wmt-1.109.

[8] **Rawat S.S., Mishra A.K.** Review of methods for handling class imbalance in classification problems. Agrawal J., Shukla R.K., Sharma S., Shieh Cs. (Eds.) Data Engineering and Applications. IDEA 2022. Lecture Notes in Electrical Engineering. Singapore: Springer; 2022; (1146):3–14. DOI:10.1007/978-981-97-0037-0_1. Available at: https://link.springer.com/chapter/10.1007/978-981-97-0037-0_1.

[9] **Al-Ansari K.** Survey on word embedding techniques in natural language processing. 2020. Available at: https://www.researchgate.net/publication/343686323_Survey_on_Word_Embedding_Techniques_in_Natural_Language_Processing. (Accessed May 20, 2025).

[10] **Tarwani K.M., Edem S.** Survey on recurrent neural network in natural language processing. International Journal of Engineering Trends and Technology. 2017; 48(6):301–304. DOI:10.14445/22315381/IJETT-V48P253. Available at: https://ijettjournal.org/archive/ijett-v48p253.

[11] **Yao L., Guan Y.** An improved LSTM structure for natural language processing. 2018 IEEE International Conference of Safety Produce Informatization (IICSPI). Chongqing, China; 2018: 565–569. DOI:10.1109/IICSPI.2018.8690387. Available at: https://ieeexplore.ieee.org/abstract/document/8690387.

[12] **Kim Y.** Convolutional neural networks for sentence classification. Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP). Doha, Qatar; 2014: 1746–1751. DOI:10.3115/v1/D14-1181. Available at: https://aclanthology.org/D14-1181.

[13] **Srivastava N., Hinton G., Krizhevsky A., Sutskever I., Salakhutdinov R.** Dropout: a simple way to prevent neural networks from overfitting. Journal of Machine Learning Research. 2014; 15(1):1929–1958. Available at: https://dl.acm.org/doi/abs/10.5555/2627435.2670313.

[14] TensorFlow: large-scale machine learning on heterogeneous distributed systems. 2015. Available at: http://download.tensorflow.org/paper/whitepaper2015.pdf.

[15] **Животова А.А., Бердоносов В.Д.** Регрессионный анализ корреляции качества машинного перевода и параметров исходного текста. Информатика и системы управления. 2023; 2(76):121–134. DOI:10.22250/18142400_2023_76_2_121.

Вычислительные технологии, 2025, том 30, № 4, с. 145–158. © ФИЦ ИВТ, 2025 ISSN 1560-7534
Computational Technologies, 2025, vol. 30, no. 4, pp. 145–158. © FRC ICT, 2025 eISSN 2313-691X

INFORMATION TECHNOLOGIES

DOI:10.25743/ICT.2025.30.4.013

Neural network models for identifying toughly translating sentences

S. I. Smagin[1,*], V. E. Oklandikov[1], T. V. Kozhevnikova[1], A. A. Zhivotova[2]

[1]Computational Center FEB RAS, 680000, Khabarovsk, Russia
[2]Komsomolsk-na-Amure State University, 681013, Komsomolsk-on-Amur, Russia
*Corresponding author: Kozhevnikova Tatyana Vladimirovna, e-mail: ktvsl@mail.ru
Received May 21, 2025, revised June 03, 2025, accepted June 11, 2025.

Abstract

Purpose. This study addresses developing and evaluating neural network-based classification models for identifying sentences that are difficult to translate using machine translation systems.

Methodology. We assemble and preprocess dataset sentences labeled by translation difficulty, apply tokenization and implement multiple neural network architectures for their classification. Three models are built: a simple recurrent network (A1) using SimpleRNN layers, a long short-term memory network (A2), and a convolutional neural network (A3) with Conv1D layers. The models are trained and tested on the dataset using standard machine learning procedures, and their classification performance is evaluated using metrics such as accuracy and F1-score.

Findings. The experimental results demonstrate that the LSTM-based architecture (A2) achieves the highest classification accuracy and $F1$-score among the proposed models, indicating its superior ability to capture complex features related to translation difficulty. All models yield satisfactory

results, however clear differences in training dynamics and final performance metrics do occur. Detailed metric values for each architecture are reported, confirming the feasibility of using neural networks for this binary classification problem.

Originality/value. A novel application of neural network classifiers to the problem of detecting translation-difficult sentences is presented. The developed dataset and models can improve pre-translation analysis and help optimize machine translation pipelines by flagging challenging inputs. The approach contributes to computational linguistics by exploring different neural architectures and offering a valuable resource for further study.

Keywords: neural network, machine learning, algorithm, machine translation, classification.

Citation: Smagin S.I., Oklandikov V.E., Kozhevnikova T.V., Zhivotova A.A. Neural network models for identifying toughly translating sentences. Computational Technologies. 2025; 30(4):145–158. DOI:10.25743/ICT.2025.30.4.013. (In Russ.)

References

1. **Way A.** Quality expectations of machine translation. Moorkens J., Castilho S., Gaspari F., Doherty S. (Eds.) Translation quality assessment. Machine translation: technologies and applications. Cham: Springer International Publishing; 2018; (1):159–178. DOI:10.1007/978-3-319-91241-7_8. Available at: https://link.springer.com/chapter/10.1007/978-3-319-91241-7_8.
2. **Jiang W.** Pre-editing for machine translation. Proceedings of the 9th Conference of the Association for Machine Translation in the Americas: Government MT User Program. Denver, USA; 2010. Available at: https://aclanthology.org/2010.amta-government.6.
3. **Zhivotova A.A., Berdonosov V.D.** Optimizational pre-editing of highly specialized Russian-language texts for its machine translation into English. Information and Mathematical Technologies in Science and Management. 2024; 2(34):169–182. DOI:10.25729/ESI.2024.34.2.016. (In Russ.)
4. **Zaki M.Z.** Revolutionising translation technology: a comparative study of transformer models — BERT, GPT, and T5. Computer Science & Engineering: An International Journal. 2024; 14(3):15–27. DOI:10.5121/cseij.2024.14302. Available at: https://www.cseij.org/papers/v14n3/14324cseij02.pdf.
5. **Šuppa M., Benešová K., Švec A.** Cost-effective deployment of BERT models in serverless environment. Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies: Industry Papers. 2021: 187–195. DOI:10.18653/v1/2021.naacl-industry.24. Available at: https://aclanthology.org/2021.naacl-industry.24.
6. **Han A.L.F., Wong D.F., Chao L.S., He L., Lu Y., Xing J., Zeng X.** Language-independent model for machine translation evaluation with reinforced factors. Proceedings of Machine Translation Summit XIV: Posters. Nice, France; 2013. Available at: https://aclanthology.org/2013.mtsummit-posters.3.
7. **Han L., Sorokina I., Erofeev G., Gladkoff S.** CushLEPOR: customising hLEPOR metric using Optuna for higher agreement with human judgments or pre-trained language model LaBSE. Proceedings of the Sixth Conference on Machine Translation. 2021: 1014–1023. Available at: https://aclanthology.org/2021.wmt-1.109.
8. **Rawat S.S., Mishra A.K.** Review of methods for handling class imbalance in classification problems. Agrawal J., Shukla R.K., Sharma S., Shieh Cs. (Eds.) Data Engineering and Applications. IDEA 2022. Lecture Notes in Electrical Engineering. Singapore: Springer; 2022; (1146):3–14. DOI:10.1007/978-981-97-0037-0_1. Available at: https://link.springer.com/chapter/10.1007/978-981-97-0037-0_1.
9. **Al-Ansari K.** Survey on word embedding techniques in natural language processing. 2020. Available at: https://www.researchgate.net/publication/343686323_Survey_on_Word_Embedding_Techniques_in_Natural_Language_Processing. (Accessed May 20, 2025).
10. **Tarwani K.M., Edem S.** Survey on recurrent neural network in natural language processing. International Journal of Engineering Trends and Technology. 2017; 48(6):301–304. DOI:10.14445/22315381/IJETT-V48P253. Available at: https://ijettjournal.org/archive/ijett-v48p253.
11. **Yao L., Guan Y.** An improved LSTM structure for natural language processing. 2018 IEEE International Conference of Safety Produce Informatization (IICSPI). Chongqing, China; 2018: 565–569. DOI:10.1109/IICSPI.2018.8690387. Available at: https://ieeexplore.ieee.org/abstract/document/8690387.

12. **Kim Y.** Convolutional neural networks for sentence classification. Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP). Doha, Qatar; 2014: 1746–1751. DOI:10.3115/v1/D14-1181. Available at: `https://aclanthology.org/D14-1181`.
13. **Srivastava N., Hinton G., Krizhevsky A., Sutskever I., Salakhutdinov R.** Dropout: a simple way to prevent neural networks from overfitting. Journal of Machine Learning Research. 2014; 15(1):1929–1958. Available at: `https://dl.acm.org/doi/abs/10.5555/2627435.2670313`.
14. TensorFlow: large-scale machine learning on heterogeneous distributed systems. 2015. Available at: `http://download.tensorflow.org/paper/whitepaper2015.pdf`.
15. **Zhivotova A.A., Berdonosov V.D.** Regression analysis of the dependence of machine translation quality on source text parameters. Information Science and Control Systems. 2023; 2(76):121–134. DOI:10.22250/18142400_2023_76_2_121. (In Russ.)