УДК 004.853

Л.Э. Чалая[1], С.Г. Удовенко[2], С.А. Гринев[3]
[1] ХНУРЭ, г. Харьков, Украина, larysa.chala@nure.ua;
[2] ХНЭУ, г. Харьков, Украина, serhii.udovenko@nure.ua;
[3] ХНУРЭ, г. Харьков, Украина, sgrinyov@gmail.com

МЕТОД НЕЙРОСЕТЕВОЙ КОРРЕКЦИИ ОШИБОК В РЕДАКТИРУЕМЫХ ЭЛЕКТРОННЫХ ТЕКСТАХ

Предложен метод анализа электронных текстов, предназначенный для обнаружения орфографических и морфологических ошибок. Метод основан на применении кодера и декодера, реализуемых с помощью рекуррентной нейронной сети, обучаемой на основе корпуса параллельных англоязычных текстов, который содержит искаженные и скорректированные предложения. Результаты тестирования подтверждают эффективность применения метода для выявления ошибок в политематических текстовых документах.

ОБРАБОТКА ЭЛЕКТРОННЫХ ТЕКСТОВ, МАШИННЫЙ ПЕРЕВОД, КОРРЕКЦИЯ ГРАММАТИЧЕСКИХ ОШИБОК, КОДЕР-ДЕКОДЕР, РЕКУРРЕНТНАЯ НЕЙРОННАЯ СЕТЬ

Введение

В последние годы в области обработки текстов на естественном языке получили развитие исследования, связанные с мониторингом социальных сетей и обработкой преднамеренно искаженных текстов. Разработаны различные методы решения задач информационного поиска, машинного перевода и т.п. [1]. При этом базовые принципы обработки, как правило, связываются с компьютерной лингвистикой, однако многие новые задачи (например, выделение ключевых слов в документах или мониторинг социальных сетей), язык которых может очень сильно отличаться от «канонического», решаются и без ее применения. К задачам, требующим применения новых методов обработки текстов, можно отнести извлечение мнений, определение эмоциональной окраски текстов, анализ реального влияния источников информации, обработку некорректных или преднамеренно искаженных текстов [2].

Важной задачей автоматического анализа электронных текстов является коррекция грамматических и морфологических ошибок (GEC) [3]. Современные поисковые системы и текстовые редакторы (Google, Word, META) частично решают эту задачу. Они содержат орфографические корректоры, которые сохраняют все формы слов и статистику ошибок. Орфокоректоры подобного типа хорошо работают в облачных вычислениях, но показывают невысокую скорость на персональных компьютерах с ограниченными вычислительными ресурсами. Здесь корректоры, предназначенные для проверки правописания, как правило, хорошо работают лишь при наличии одиночных ошибок в словах, содержащихся в словаре. Проверка текста в системах анализа и коррекции ошибок в текстах может вестись в режиме «off-line», когда формируется протокол замечаний по тексту, либо в режиме «on-line», когда исправление ошибок ведется по мере их обнаружения (как правило, после получения соответствующего подтверждения от пользователя). При обнаружении ошибки система может предложить вариант ее исправления, а при наличии нескольких вариантов – их упорядоченный список. Замечания по тексту также могут носить различный характер. Они могут быть локальными (указывается фрагмент текста с ошибкой) и глобальными (выдается диагностическое сообщение, касающееся всего текста).

Следует отметить, что в настоящее время не существует универсальных систем редактирования черновых электронных документов. Эффективность применения существующих методов коррекции ошибок в системах электронных текстов на естественном языке зависит от характера конкретных задач, решаемых в процессе такой обработки. При этом имеют большое значение язык и тематическая направленность текста, а также тип исправляемых ошибок, формат представления текста, трудоемкость алгоритма коррекции и требования к качеству коррекции.

Одной из важных практических задач, связанных с необходимостью коррекции ошибок и опечаток в черновых электронных текстах, является задача анализа документов на иностранных языках, составленных пользователями, не являющимися носителями этих языков. Как правило, такие документы содержат большое количество грамматических и морфологических ошибок и нуждаются в редактировании. Для решения задач автоматического редактирования таких документов перспективным является применение машинного обучения и специализированных нейросетевых моделей [4, 5]. В последнее время были разработаны нейронные моделей коррекции языка (NLC), показывающие обнадеживающие результаты [6]. Такие модели могут быть эффективно использованы в задачах GEC, поскольку они позволяют исправлять ошибочные фразы и отдельные грамматические ошибки, которые не были замечены в подготовительном наборе. Путем внедрения NLC в системы GEC достигается значительное повышение

качества исправления грамматических ошибок в исходных электронных текстах на этапе их предварительного редактирования.

Целью настоящей работы является исследование эффективности использования нейронных сетей в системах коррекции электронных англоязычных текстов с целью исправления грамматических и морфологических ошибок и разработка модифицированного метода с использованием нейросети многослойного персептрона.

Достижение этой цели обусловило необходимость решения следующих задач: анализ эффективности различных подходов к проблеме коррекции электронных текстов; имитационное моделирование различных архитектур нейронных моделей редактирования электронных текстов; разработка и тестирование модифицированного метода коррекции ошибок в электронных англоязычных текстах.

1. Анализ задачи автоматической коррекции ошибок в редактируемых электронных текстах

В последнее время повысился интерес к решению практических задач коррекции ошибок различных типов в черновых англоязычных электронных текстах.

В частности, это связано с существенным ростом числа людей, изучающих английский как второй язык (English as a Second Language-ESL) во всем мире.

Рассмотрим существующие GEC-подходы к обработке редактируемых англоязычных документов, которые могут быть полезны при разработке рекомендаций и учебных пособий для пользователей ESL-систем.

Эти походы позволяют создать программы проверки англоязычных текстов, подготовленных на естественном языке (ЕА) и предположительно содержащих значительное количество грамматических и морфологических ошибок. Такие программы либо исправляют входной текст автоматически, либо формируют некоторый протокол замечаний. Любой ЕЯ представляет собой сложную систему с множеством правил и внутренних связей. Точность и правильность работы программ GEC определяется глубиной анализа редактируемых текстов. Достаточно глубокий анализ может быть достигнут только для определенных узких предметных областей (вследствие специфичности подъязыков таких областей, связанной с наличием в каждой области своих терминов и семантических отношений).

Для создания универсальных GEC-систем, работающих с ЕЯ без потери глубины анализа, в настоящее время не хватает либо технических возможностей (быстродействия, памяти), либо теоретической базы (например, схемы достаточно полного, глубокого и непротиворечивого описания семантики ЕЯ). Однако в GEC-системах, предназначенных для ESL-пользователей и разных предметных областей, может быть принята концепция поверхностного анализа редактируемых текстов (ПАРТ), осуществляемого в реальном масштабе времени (например, в процессе электронной переписки ESL-пользователей). Дальнейшее усложнение функциональных возможностей и повышение качества GEC-систем в практических областях возможно с помощью применения в таких системах более сложных (с точки зрения учета особенностей ЕЯ) лингвистических моделей. Методы обнаружения и коррекции орфографических и морфологических ошибок в электронных текстах широкой тематики базируются на представлении о тексте как о цепочке независимо появляющихся словоформ.

Большая часть методов ПАРТ, которые используются во время редакторской GEC-обработки текстовой информации, сводится к задаче поиска по сходству соответствующего эталонного слова.

Наиболее распространенными методами словарного поиска по сходству являются: последовательное сканирование словаря; расширение выборки; применение n-грамм; хеширование; триангуляционные деревья; trie-деревья (лучи) и т.д. [7].

При поиске по сходству с использованием полного перебора всех терминов словаря, или последовательного сканирования, необходимо каждое слово из словаря сравнить с образцом. В большом словаре многократное повторение даже простого сравнения требует значительных затрат времени. При этом основной проблемой является значительная вычислительная сложность, что делает неэффективным его использование для поиска на больших массивах данных. Идея метода расширения выборки заключается в том, чтобы свести задачу к поиску точного равенства путем построения наиболее вероятных «ложных» вариантов поискового шаблона и поиска всех таких «ошибочных» вариантов в словаре. При этом шаблон содержит слова, для которых расстояние редактирования мало, а при допущении двух и более ошибок расширенная выборка становится слишком большой. Кроме обычных ошибок, этот метод может учитывать изменение формы (окончание, суффикс) слова. Метод расширения выборки широко используется программами проверки орфографии при ПАРТ-анализе.

Для точного поиска слов по сходству традиционно используются деревья и хэш-таблицы [7]. Идея, которая лежит в основе хэш-таблиц, заключается в том, что множество слов можно отобразить на ограниченном множестве целых чисел X: $x \in X, 1 \le x \le n$. Хеш-таблица является массивом размерности n. Чтобы сохранить слово S, надо вычислить хеш-функцию $i = h(S)$ и присвоить i-му элементу массива значение S.

ПАРТ-методы, основанные на построении различных вариантов деревьев, считаются достаточно быстрыми. Существуют различные типы деревьев: деревья, строящиеся на множестве элементов,

которые можно сравнивать (обычные бинарные деревья, b-деревья, b+-деревья и т.д.); деревья, строящиеся на множестве элементов, которые можно представить в виде последовательности (trie-деревья); деревья, в которых задана метрика на множестве элементов (триангуляционные деревья); частотные деревья и kd-деревья, которые за счет особой структуры данных позволяют уменьшить количество сравнений.

Наиболее популярными деревьями, которые используются для поиска слов по сходству, является триангуляционные и trie-деревья. Trie-деревья, в отличие от обычных сбалансированных деревьев, имеют общее начало для всех слов, расположенных в одном поддереве. Каждое ребро дерева помечено некоторой строкой. Терминальным вершинам («листьям») соответствуют строки списка. Обычно trie-деревья используются для поиска по подстроке, но их можно эффективно использовать и для поиска по сходству. Чтобы обеспечить поиск по подстроке, необходимо хранить в trie-дереве все суффиксы (или префиксы терминов). Триангуляционные деревья позволяют индексировать множество произвольных структур, при условии, что на них задана метрика.

Для уменьшения множества данных, подлежащих ПАРТ-обработке, применяется предварительная фильтрация. Она проста в реализации и с помощью быстрых и неточных функций сравнения позволяет существенно уменьшить выборку вероятно-релевантных записей. К методам предварительной фильтрации можно отнести методы снижения размерности данных, или методы, которые реализуют быстрые и приблизительные оценки известных строчных метрик. Наиболее применяемым методом предварительной фильтрации текстов при ПАРТ-обработке является использование *n*-грамм, в соответствии с которым для описания слова строится набор подстрок размера n, который может быть или фиксированным для всех подстрок, или оптимизироваться для каждой подстроки отдельно. Например, слово $S = \{s_1, s_2, ..., s_m\}$ будет описывать набор n-грамм $\{s_1 s_2 ... s_n, s_2 s_3 ... s_{n+1}, ..., s_{n-m+1} ... s_{m-1} s_m\}$.

При использовании n-граммного индекса словарь содержит все *n*-граммы, построенные из всех слов словаря. Каждый инвертированный список ставит в соответствие *n*-грамме все слова словаря, которые содержат данную n-грамму. Если необходимо найти слова, содержащие n-граммы образца, поиск проводится по всем *n*-граммам словаря, которые с ними совпадают. При поиске может учитываться позиция *n*-грамм в слове, если это было учтено при построении индекса. Результаты проведения поиска на каждой *n*-грамме оцениваются с помощью коэффициента оценки пересечения множеств. Если такой коэффициент больше заданного порогового значения, то слово словаря считается значимым и добавляется к результатам. Далее полученное множество может обрабатываться более точным методом поиска сходства для получения окончательного результата. К недостаткам всех рассмотренных методов поиска по сходству можно отнести сложность построения их хорошо сбалансированных параллельных аналогов.

Существуют также и другие подходы к обнаружению и исправлению грамматических ошибок в англоязычных текстах. Некоторые из них основаны на классификации по набору классифицирующих модулей, каждый из которых адресован конкретному типу ошибки, и статистическом машинном переводе (SMT), решающем задачу перевода с «плохого» на «хороший» английский язык. Гибридные модификации этих подходов сочетают в себе процедуры классификации и SMT, а также частично используют компоненты, основанные на правилах [8]. Каждый подход имеет свои особенности. Поскольку классификационный подход способен сосредоточиться на каждом отдельном типе ошибки, используя отдельный классификатор, то он лучше работает на типах ошибок, для которых можно построить специализированные классификаторы (например, для ошибок типа связка «субъект-глагол»). Таким образом, недостаток классификационного подхода заключается в том, что один классификатор должен быть построен для каждого типа ошибок, поэтому комплексная система GEC нуждается в построении множества классификаторов, что усложняет ее конструкцию. Кроме того, классификационный подход не адресуется нескольким типам ошибок, которые могут взаимодействовать.

Применение SMT подхода позволяет естественным образом учитывать взаимодействия между словами в предложении и находить наилучшее скорректированное предложение. Методы ПАРТ-обработки, основанные на SMT, дают возможность осуществлять преобразование текста с ошибками в исправленный текст без явного моделирования типов ошибок.

Недостатком этого подхода является необходимость наличия на этапе обучения большого параллельного обучающего корпуса, состоящего из текстов, созданных ESL-пользователями, и соответствующих исправленных текстов.

Общая структура коррекции ошибок при ПАРТ-обработке англоязычных электронных текстов во многом зависит от учитываемых типов ошибок. К таким ошибкам следует отнести:

– орфографические ошибки (пропуск одной буквы, замена одной буквы, перестановка двух рядом стоящих букв, одна лишняя буква);

– морфологические ошибки (ошибки в окончаниях, употребление отсутствующих в языке форм слов, несоблюдение правил их чередования в предложении, употребление незнакомых вариантов слов, редких и жаргонных терминов);

– синтаксические ошибки (пунктуационные ошибки, нарушение нормативного порядка слов, некорректное использование пробелов;

– лексико-семантические ошибки (употребление слов в ненормативном значении, нарушение лексической сочетаемости, семантические противоречия).

При формировании электронных текстов ESL-пользователями могут возникать ошибки всех рассмотренных типов. Примеры типичных ошибок, содержащихся в исходных текстах, подлежащих GEC – коррекции, приведены в табл. 1.

Таблица 1

Примеры типичных ошибок и применения редких слов в ESL-текстах

Фрагмент текста	Комментарий к ошибке
visi**tt**ed	visi**t**ed (удвоение)
basic**l**y	basic**al**ly (пропуск)
all for **not**	all for **naught** (значение фразы)
a**ht**lete	a**th**lete (перестановка)
c**a**me	c**o**me (замена)
17.06.12	Редкое слово
;)	Смайлик (редкое слово)
on the same token	**by** the same token (неправильное употребление глагола)
The woman **which**	The woman **who** (неправильное наречие)

2. Нейросетевые модели представления и коррекции электронных текстов

Рассмотрим некоторые варианты применения искусственных нейросетей в задачах исправления ошибочных фраз и отдельных грамматических ошибок при обработке англоязычных электронных текстов. Нейросетевые модели позволяют использовать распределенные векторные представления слов (distributed vector space word representations). В [4] и [5] были предложены две нейросетевых языковых модели с таким представлением слов – Continuous Bag-of-Words и Skip-gram. Основным преимуществом этих моделей являются существенно меньшие вычислительные затраты на обучение по сравнению с другими известными нейросетевыми языковыми моделями. Это достигается отчасти благодаря использованию иерархического программного обеспечения, основанного на представлении слов словаря в виде дерева Хаффмана.

Данные модели реализуются при помощи двухслойной либо трехслойной нейронной сети. Распределенные векторные представления слов заключены в синаптических весах первого слоя сети. Рассмотрим архитектуру сети для модели Continuous Bag-of-Words.

В этой модели слово w_t предсказывается нейронной сетью по его контексту $w_{t-k},...,w_{t-1},w_{t+1},...w_{t+k}$. Особенностью этой модели является динамический размер окна: число I принимает равновероятные значения от 1 до некоторого *N*. Обучение заключается в максимизации целевой функции следующего вида:

$$L=\sum_{t,k}\log P(w_t \mid w_{t-k},...,w_{t-1},w_{t+1},...,w_{t+k}). \quad (1)$$

Второй (скрытый) слой сети трехслойной нейронной сети, соответствующей этой модели, предназначен для усреднения распределенных векторов, соответствующих словам контекста. Число нейронов в нем равно размерности D распределенных векторов.

Каждой нелистовой вершине дерева Хаффмана, построенного по словарю V, соответствует один нейрон третьего слоя с D синаптическими весами. Пусть X – множество нейронов третьего слоя сети, соответствующих всем нелистовым вершинам дерева Хаффмана на пути от корню к слову w_t. Каждый нейрон из X осуществляет скалярное умножение вектора своих синаптических весов на вектор выходных сигналов второго слоя (т.е. среднее по векторам контекста), а к результату применяет логистическую функцию. Совокупность выходных сигналов w_t^* нейронов X (их количество равно длине кода Хаффмана слова w_t) сравнивается с кодом Хаффмана слова w_t. Целью обучения является минимизация функции следующего вида:

$$P(w_t \mid w_{t-k},...,w_{t-1},w_{t+1},...,w_{t+k})=\prod_i |\, w_{t_i}-w_{t_i}^* \,|, \quad (2)$$

где $w_{t_i}\in\{0,1\}$ – цифра в разряде i кода Хаффмана для слова w_t.

После вычисления $w_{t_i}^*$ производится коррекция распределенных векторов слов контекста (синаптических весов определенной области нейронов первого слоя), а также синаптических весов нейронов для X в направлении возрастания данной условной вероятности.

Основное отличие двухслойной модели Skip-gram от модели Continuous Bag-of-Words заключается в том, что слово w_t предсказывается столько раз, сколько слов в его контексте (на основе только одного из слов контекста).

Можно показать, что вычислительная сложность обучения для модели Continuous Bag-ofWords составляет $Q_1=O\left(N\times D+D\times log_2|V|\right)$, а для модели Skip-gram – $Q_2=O\left(N\times D+N\times D\times log_2|V|\right)$.

Нейросетевые модели коррекции ошибок при ПАРТ-обработке исходных англоязычных текстах могут быть представлены в рекуррентных вариантах, что дает дополнительные преимущества при реализации вычислительных GEC - процедур.

3. Модифицированный метод нейросетевой коррекции ошибок в электронных текстах

Предлагаемый метод коррекции ошибок на базе естественного языка основан на применении кодера и декодера, реализуемых с помощью рекуррентной нейронной сети (RNN), обучаемой на основе корпуса параллельных текстов, который содержит искаженные и скорректированные предложения. Пример работы такого метода приведен на рис. 1.

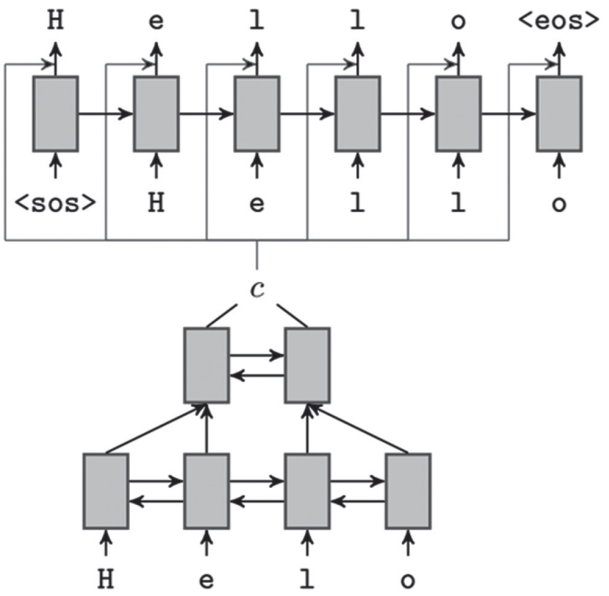

Рис. 1. Пример применения кодера и декодера RNN-модели

Искаженное входное предложение (на рис.1 – «Helo») обрабатывается системой коррекции, после чего на выходе формируется исправленное предложение (на рис. 1 – Hello»). Таким образом, модель коррекции состоит из кодера и декодера, реализуемых с помощью нейросети с пирамидальной двунаправленной RNN архитектурой, рассмотренной в работе [9]. Декодер использует так называемый «механизм внимания» для установления связи с закодированным представлением и посимвольной генерации выходного предложения.

Модель нейронной сети работает на уровне символов, как в кодировщике, так и в декодере. Это связано с тем, что известные нейронные модели машинного перевода на уровне слов с фиксированным словарем плохо подходят для обработки фрагментов, содержащих, например, многоразрядные числа, смайлики, веб-адреса и т.п. Несмотря на более длинные последовательности в модели на основе символов, работа системы коррекции не усложняется существенно, поскольку большая часть выполняемых здесь операций сводится к копированию символов из источника в целевой текст.

С учетом входного вектора x_t, прямые, обратные и комбинированные активации j-го скрытого слоя рассчитываются как:

$$\begin{aligned} f_t^{(j)} &= GRU(f_{t-1}^{(j)}, c_t^{(j-1)}), \\ b_t^{(j)} &= GRU(b_{t+1}^{(j)}, c_t^{(j-1)}), \\ h_t^{(j)} &= f_t^{(j)} + b_t^{(j)}, \end{aligned} \qquad (3)$$

где GRU – функция краткосрочной памяти рекуррентного блока RNN.

При $j = 0$ $c_t^{(0)} = x_t$,

Входные данные предыдущего входного слоя ($j > 0$):

$$c_t^{(j)} = \tanh(W_{pyr}^j [h_{2t}^{(j-1)}, h_{2t+1}^{(j-1)}]^T + b_{pyr}^j). \qquad (4)$$

Матрица весов *Wpyr* вдвое сокращает количество скрытых состояний для каждого дополнительного скрытого слоя, и, следовательно, кодировщик имеет пирамидальную структуру. В заключительном скрытом слое формируется закодированное представление входного предложения.

Сеть декодера является рекуррентной нейронной сетью, содержащей рекуррентные блоки с M скрытыми слоями. После конечного скрытого слоя сеть задает условие для закодированного представления результата, используя механизм внимания.

На j-ом слое декодера скрытые активации вычисляются следующим образом:

$$d_t^{(j)} = GRU(d_{t-1}^{(j)}, d_t^{(j-1)}), \qquad (5)$$

где $d_t^{(M)}$ – значение выходного скрытого слоя.

После этого взвешенную сумму закодированных скрытых состояний связывают с $d_t^{(M)}$, после чего получают окончательный выходной слой результирующей функции.

Для того, чтобы отфильтровать ложные правки, в предлагаемом методе использован классификатор редактирования правок. Запускается классификатор на нескорректированных предложениях из обучающих данных для генерации потенциально исправленных предложений. Затем согласовывается потенциальные предложения с нескорректированными путем минимизации расстояния Левенштейна на уровне слова между каждым потенциальным и неисправленным предложением. Расстояние Левенштейна выражается здесь следующими рекуррентными соотношениями:

$$d_L(a_1 \ldots a_n a_{n+1}, b_1 \ldots b_m b_{m+1}) = \min \begin{cases} d_L(a_1 \ldots a_n, b_1 \ldots b_m b_{m+1}) + 1 \\ d_L(a_1 \ldots a_n a_{n+1}, b_1 \ldots b_m) + 1 \\ d_L(a_1 \ldots a_n, b_1 \ldots b_m) + 1 \end{cases},$$

где учитывается возможность осуществления таких операции коррекции, как удаление символа a_{n+1}, вставки элемента b_{m+1}, изменение элемента a_{n+1} на b_{m+1} при $a_{n+1} \neq b_{m+1}$.

Для упрощения вычислительной процедуры в классификаторе используется нормированное представление расстояний Левенштейна:

$$d_{L1}(a_1 \ldots a_n, b_1 \ldots b_m) = \frac{d_L(a_1 \ldots a_n, b_1 \ldots b_m)}{\max(n, m)}. \qquad (6)$$

Для формирования редакционных предписаний будем использовать алгоритм Хиршберга [11]. В соответствии с этим алгоритмом идея рекурсивного отображения последовательности $a_1 \ldots a_i$ в последовательность $b_1 \ldots b_{j^*}$ основывается на следующих равенствах:

$$d_L(a_1 \ldots a_n, b_1 \ldots b_m) = d_L(a_n \ldots a_1, b_m \ldots b_1)$$

$$d_L(a_1 \ldots a_i a_{i+1} \ldots a_n, b_1 \ldots b_m) =$$

$$= \min_{j \in 1, \ldots, m-1} (d_L(a_1 \ldots a_i, b_1 \ldots b_j) + d_L(a_{i+1} \ldots a_n, b_{j+1} \ldots b_m)).$$

Эти равенства позволяют построить вычислительные процедуры алгоритма Хиршберга для

различного соотношения количества элементов во входной и выходной последовательностях.

Смежные сегменты предложений, которые не совпадают после реализации этого алгоритма, извлекаются в процессе предложенных правок.

При нейросетевой реализации схемы «кодер-декодер» входное предложение произвольной длины должно быть сжато в вектор фиксированного размера. Этот подход кажется довольно странным, учитывая то, Кроме того, качество обработки существенно снижается, если модель «кодер-декодер» имеет малый размер. Таким образом, для успешной обработки длинных предложений мощность кодера должна быть значительной. Топология модели, реализующей алгоритм коррекции, и ее алгоритм обучения должны быть такими, чтобы данные большой размерности можно было бы передавать со входа нейронной сети на ее выходы через канал сравнительно небольших размеров. Для реализации сжатия такого рода может использоваться многослойный персептрон следующей архитектуры: количество нейронов во входном и выходном слое одинаково и равно размерности сжимаемых данных; между этими слоями располагаются один или более промежуточных слоев меньшего размера. Число промежуточных слоев определяет степень сложности преобразования данных.

Требованиям, необходимым на реализации процедур сжатия данных в процессе коррекции, удовлетворяет автоассоциативный трехслойный персептрон типа «Бутылочное горлышко», структура которого приведена на рис.2. [11].

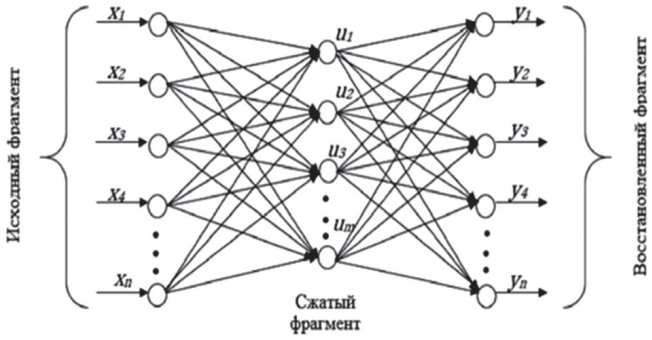

Рис. 2. Многослойный персептрон с архитектурой типа «бутылочное горлышко»

Обучение такого персептрона может быть осуществлено с помощью любой процедуры обратного распространения ошибок с тем отличием, что в качестве обучающего образа $d(k)$ используется сам входной сигнал $x(k)$, подлежащий сжатию. Отметим, что на нулевой слой поступает n-мерный вектор входных сигналов $x(k)$ ($n_0 = n$), первый скрытый слой содержит $n_1 = n$ нейронов, второй скрытый слой n нейронов и выходной слой – $n_3 = n$ нейронов. Целью ассоциативного обучения является восстановление на выходе сети сигнала, наилучшим образом аппроксимирующего входной сигнал $x(k)$. Сжатие информации происходит во втором скрытом слое, содержащем меньшее число нейронов, чем первый и выходной слои. Именно с выхода второго скрытого слоя снимается «сжатый» сигнал. Отметим, что такая архитектура имеет преимущество перед обычными многослойными персептронами по скорости обучения.

4. Результаты тестирования

Для проведения тестирования использовался словарь, который включал 98 символов: набор печатных символов ASCII, специальных символов <sos>, <eos> и <unk>, обозначающих начало, конец предложения и неизвестные символы соответственно. В эксперименте был использован набор данных, который был получен из набора общих задач CoNLL, который содержит около 80 тысяч предложений из рефератов, написанных учащимися, изучающими английский язык с исправлениями и описаниями типов ошибок. На рис. 3 приведен пример экранной формы с результатом коррекции.

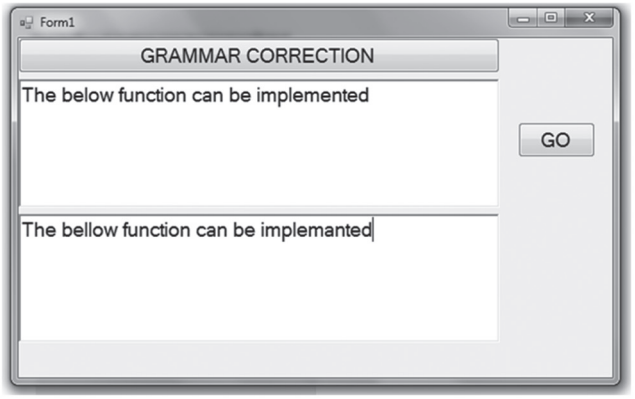

Рис. 3. Пример экранной формы с результатом коррекции

Результаты эксперимента свидетельствуют о приемлемом качестве исправления грамматических и морфологических ошибок при применении модифицированного нейросетевого RNN - метода. В частности, из 345 ошибок и опечаток, связанных с отсутствием, вставкой и заменой букв в исходных тестах, были обнаружены и исправлены 332 ошибки.

Выводы

В данном докладе исследована эффективность использования нейронных сетей в системах анализа англоязычных электронных текстов с целью исправления грамматических и морфологических ошибок и разработана модификация GEC -метода с использованием рекуррентной нейронной сети. Предложенный подход к коррекции ошибок на базе естественного языка, основан на применении нейросетевой реализации кодера и декодера, обучаемых на основе корпуса параллельных текстов, содержащего правильные и неправильные предложения.. Кодер сопоставляет входное предложение с представлением более высокого уровня с пирамидальной двунаправленной RNN архитектурой.

Декодер также является рекуррентной нейронной сетью, которая использует механизм внимания на основе содержимого кодера для установления связи с закодированным представлением и посимвольной генерации выходного предложения. Для фильтрации ложных правок был использован специальный классификатор редактирования. Предварительно отредактированные предложения согласовываются с исходными путем минимизации расстояния Левенштейна.

Перспективным развитием метода является проведение экспериментов по усовершенствованию схемы нейросетевой коррекции путем учета более сложных типов ошибок и применения комбинированных нейросетевых архитектур GEC-систем.

Список литературы:

1. *Lizunov P.* Near-dublicate detection for tables on the base of the local-sensitive hashing and the nearest neighbor search methods / . Lizunov, A. Biloshcytskyi, A. Kuchansky, S. Biloshchytska, L. Chala // – Eastern-European Journal of enterprise technologies. – 2016 – №6 – P. 4– 10. **2.** *Sutskever I.* Sequence to Sequence Learning with Neural Networks [Text] / I. Sutskever, O. Vinyals, Q. V.Le // In Advances in Neural Information Processing Systems. – 2014. – P. 3104–3112. **3.** *Luong T.* Addressing the Rare Word Problem in Neural Machine Translation / T. Luong, I. Sutskever, Q Le // – In Proceedings of the ACL-IJCNLP. – 2013. – P. 11 – 19. **4**. *Mikolov T.* Efficient estimation of word representations in vector space / T. Mikolov, K.. Chen, G. Corrado, J. Dean // In Proceedings of Workshop at ICLR. – 2013. – 12p. **5**. *Mikolov T.* Distributed representations of words and phrases and their compositionality / T. Mikolov, I. Sutskever, K. Chen //. – In Proceedings of NIPS. – 2013. – 9 p. **6**. *Kalchbrenner N.* Recurrent Continuous Translation Models / N Kalchbrenner, P Blunsom // – In Proceedings of the Conference on Empirical Methods in Natural Language Processing. – 2013. – P. 1700 – 1709. **7**. *Бойцов Л.М.* Использование хеширования по сигнатуре для поиска по сходству / Л.М. Бойцов // Прикладная математика и информатика. – М.: МГУ. – 2001. – № 8. – С. 135 – 154. **8.** Russo, L. Approximate String Matching with Compressed Indexes / L. Russo, G. Navarro, A. Oliveira, P. Morales // Algorithms. – 2009. – P. 1105 – 1136. **9**. Bahdanau, D. Neural Machine Translation by Jointly Learning to Align and Translate [Text]/ D. Bahdanau, K. Cho, Y. Bengio //CoRR, abs/1409.0473, – 2014. – 15 p. **10**. Hirschberg D. S. A linear space algorithm for computing maximal common subsequences / D. S. Hirschberg // Communications of the ACM. – 1975. –. Vol. 18, no. 6. P. 871–883. **11**. Ham, F. M. Principles of Neurocomputing for Science & Engineering. [Text]/ F. M. Ham, I. Kostanic– N.Y.: Mc Graw-Hill, Inc., 2001. – 642 p.

Поступила в редколлегию 23.06.2017