Московский государственный университет имени М.В. Ломоносова

Факультет вычислительной математики и кибернетики

Кафедра математических методов прогнозирования

Чиркова Надежда Александровна

Байесовская регуляризация рекуррентных нейронных сетей

МАГИСТЕРСКАЯ ДИССЕРТАЦИЯ

Научный руководитель:

Д. А. Кропотов

Научные консультанты:

к. ф.-м. н. Д. П. Ветров

Е. М. Лобачева

Москва, 2018

Содержание

1 Введение

Рекуррентные нейронные сети показывают высокие результаты в различных задачах, связанных с анализом и генерацией последовательных данных: при обработке естественного языка, распознавании речи, создании вопросно-ответных систем [3, 1, 8, 26] и др. Этот эффект во многом обусловлен наличием большого количества настраиваемых по данным параметров нейронной сети. В частности, для сложных задач, таких как машинный перевод [26] или распознавание речи [1], современные рекуррентные архитектуры содержат миллионы параметров. Модели с большим числом параметров склонны к переобучению, поскольку могут запомнить целевые метки на объектах обучающей выборки без сохранения обобщающей способности для работы на тестовых объектах.

Кроме того, подобные модели требуют большого объема памяти для хранения параметров. Это допустимо для работы на сервере, однако делает невозможным использование рекуррентных нейронных сетей на портативных устройствах, например, на смартфонах. Поэтому становится актуальной задача регуляризации и сжатия рекуррентных нейронных сетей, то есть уменьшения числа параметров нейросети без потери качества работы результирующего предсказывающего алгоритма. Сжатие может ускорить работу алгоритма предсказания, что также является актуальным вопросом при использовании нейросетевых моделей в системах реального времени.

В литературе известны различные способы регуляризации нейронных сетей, применимые, в частности, для рекуррентных нейронных сетей: ранний останов во время обучения, L_1 и L_2-регуляризация [7], дропаут [6], нормализация [15]. Значительный недостаток этих способов регуляризации — необходимость настройки гиперпараметров для конкретной задачи, например, коэффициента регуляризации для L_2-регуляризации или вероятности сохранения нейрона для дропаута. С другой стороны, существует группа теоретически обоснованных методов, называемых байесовской регуляризацией, позволяющих регуляризовать нейронную сеть, а также удовлетворить дополнительные требования к модели без настройки дополнительных гиперпараметров.

В рамках байесовского подхода [5, 6] параметры рекуррентной нейронной сети воспринимаются как случайные величины. В ходе обучения априорное распределение на параметры, отображающее исходные ожидания о величинах параметров, трансформируется в апостериорное распределение, обусловленное на данные. Результирующий алгоритм предсказания представляет собой ансамбль рекуррентных нейронных сетей, качество работы которого может значительно превышать качество работы отдельной

нейронной сети. Некоторые техники регуляризации нейронных сетей были интерпретированы с точки зрения байесовского подхода [17, 2], что позволило усовершенствовать метод регуляризации и обогатить нейросетевую модель новыми свойствами. В частности, в [17] описывается разреживающий вариационный дропаут — метод сжатия нейронных сетей прямого распространения, основанный на вариационном дропауте [12], байесовской интерпретации дропаута. В [20, 4] предложены расширения этого подхода для получения групповой разреженности, позволяющие удалять веса группами, тем самым ускоряя проход вперед по сети нейросети. Однако данные методы не были ранее применены к рекуррентным нейронным сетям.

Целью данной работы является разработать метод байесовской регуляризации и разреживания рекуррентных нейронных сетей, позволяющий уменьшить число параметров модели и ускорить проход вперед по нейросети без значительной потери качества работы алгоритма. Для достижения поставленной цели необходимо решить следующие задачи:

1. адаптировать метод разреживающего вариационного дропаута к рекуррентным нейронным сетям с учетом их особенностей;

2. разработать метод группового разреживания рекуррентных нейронных сетей с учетом специфики современных гейтовых рекуррентных архитектур;

3. провести эксперименты по изучению свойств разработанных методов и по сравнению методов с существующими подходами.

Среди описанных в литературе подходов к сжатию рекуррентных нейронных сетей можно выделить три большие группы: основанные на разложении матрицы весов [24, 14], понижении точности [10] и разреживании [18], то есть обнулении весов. Методы первой группы аппроксимируют матрицы весов рекуррентной нейронной сети с помощью низкоранговых матричных или тензорных разложений. Например, в [24] используется разложение тензорного поезда (TT-разложение), а в [14] — произведение Кронекера. Методы второй группы подразумевают хранение параметров модели в виде чисел меньшей точности, что экономит общую память на хранение весов. В частности, в [10] веса и активации ограничиваются бинарными величинами, и предлагается способ вычисления градиентов в такой модели. Наконец, методы третьей группы постепенно обнуляют часть весов на этапе обучения модели, дообучая оставшиеся ненулевые параметры. Наиболее популярен и прост среди этих методов прунинг, в котором постепенно

удаляются веса, по модулю меньшие некоторого порога. В [18] порог выбирается, исходя из нескольких гиперпараметров, контролирующих частоту и долю обнуления весов. Существуют также методы группового прунинга, удаляющие веса группами. В [25] в группы объединяют все веса, относящиеся к одному скрытому нейрону, и используются $L_{1/2}$ регуляризацию по этим группам. Группы с маленькой L_2-нормой весов обнуляют. Метод, предлагаемый в данной работе, относится к третьей группе.

Основная часть работы организована следующим образом. В главе 2 приводится формальная постановка задачи. Далее в главе 3 описываются подходы, на которые опирается разработанный метод, а в главе 4 описывается сам метод. В главе 5 приводятся эксперименты, демонстрирующие работу метода.

2 Постановка задачи

2.1 Определения и обозначения

Для конкретики в обозначениях в данной работе будут рассматриваться задачи обучения с учителем, хотя излагаемые методы могут применяться и в задачах обучения без учителя. В задаче обучения с учителем по выборке пар объект–целевая переменная $\mathcal{D} = \{(x_1, y_1), \ldots, (x_N, y_N)\}$, $y_i \in \mathbb{Y}$, $x_i \in \mathbb{R}^n$ необходимо построить предсказывающий алгоритм $a : \mathbb{R} \to \mathbb{Y}$, минимизировав функционал качества

$$\sum_{i=1}^{N} L(y_i, a(x_i)) \to \min_a,$$

где L — функция потерь, показывающая величину ошибки предсказания на одном объекте. К примеру, в задаче классификации $\mathbb{Y} = \{1, \ldots K\}$, в задаче регрессии — $\mathbb{Y} = \mathbb{R}$. Часто алгоритм задают в виде параметрического семейства $a(x, w)$, и затем в процессе обучения модели с помощью градиентных стохастических оптимизационных методов находят оптимальные с точки зрения функционала качества параметры $w \in \mathbb{R}^D$. Модель может определяться также гиперпараметрами — величинами, которые нельзя обучить градиентными оптимизационными методами и для настройки которых необходимо применять более сложные методы.

Одно из популярных параметрических семейств — нейронные сети, позволяющие моделировать сложные зависимости в данных. Нейронная сеть задается как композиция преобразований вещественных многомерных пространств, называемых слоями нейронной сети. Одним из наиболее популярных слоев нейронных сетей является полносвязный слой:

$$h = g(Wx + b),$$

$W \in \mathbb{R}^{k \times n}$, $b \in \mathbb{R}^k$ — параметры полносвязного слоя, $h \in \mathbb{R}^k$ — скрытое представление объекта, получаемое в результате применения слоя, g — одномерная нелинейная функция, применяемая к вектору покомпонентно, например, $g(z) = \frac{1}{1+\exp(-z)}$. Элементы скрытого представления чатсо называют нейронами.

Для работы с объектами, представляющими собой последовательности произвольной длины $x = [x_0, \ldots, x_T]$, $x_t \in \mathbb{R}^n$, например текстами, часто используют рекуррентные слои. Рекуррентный слой преобразует последовательность входных элементов в последовательность скрытых состояний $[h_0, \ldots, h_T]$, $h_t \in \mathbb{R}^m$:

$$h_t = f_h(x_t, h_{t-1}), \quad t = 1 \ldots T, \quad h_0 = 0$$

В простейшем случае рекуррентный слой задается преобразованием

$$h_t = g(W^x x_t + W^h h_{t-1} + b),$$

где $W^x \in \mathbb{R}^{m \times n}, W^h \in \mathbb{R}^{m \times m}, b \in \mathbb{R}^m$ — параметры рекуррентного слоя, g — одномерная нелинейная функция, применяемая к вектору покомпонентно. На практике часто используют более сложные рекуррентные архитектуры, которые будут описаны в работе позже.

Чтобы применить рекуррентный слой к последовательности элементов неупорядоченного множества $[x'_0, \ldots, x'_T]$, $x_i \in \{1, \ldots, C\}$, например слов словаря или букв алфавита, используют слой представлений. Слой представлений задается матрицей параметров $W^e \in \mathbb{R}^{C \times p}$ и для каждого элемента последовательности возвращает строку матрицы с соответствующим номером:

$$[x'_0, \ldots, x'_T] \to [w^e_{x'_0}, \ldots, w^e_{x'_T}].$$

Под рекуррентной нейронной сетью (РНН) будем понимать нейронную сеть, преобразующую входную последовательность произвольной длины в целевую переменную, которая может быть как вектором фиксированной длины (для задач классификации или регрессии), так и вектором той же длины, что и входной объект (например, в задаче моделирования естественного языка). Параметры РНН — совокупность параметров всех слоев. Проход вперед по РНН состоит в преобразовании входной последовательности в целевую переменную. Проход назад — в вычислении градиентов функционала качества по параметрам РНН.

Классическая архитектура рекуррентной нейронной сети для обработки текстовой строки (последовательности слов) состоит из слоя представлений, подаваемого на вход рекуррентному слою, последнее скрытое состояние которого с помощью полносвязного слоя преобразуется в предсказание из множества $\mathbb{Y}$.

2.2 Задача разреживания рекуррентных нейронных сетей

Наибольшая доля параметров рекуррентной нейронной сети сосредоточена в матрицах параметров W, на долю свободных членов приходится лишь их малая часть. Аналогично наиболее вычислительно емкая часть прохода вперед — это вычисление матричных произведений.

Добиться уменьшения числа параметров РНН можно, заложив в модель требование, чтобы наибольшая часть элементов матриц параметров была равна нулю. Тогда

матрицы параметров можно будет хранить в компактном разреженном виде.

Если разреживание матриц параметров будет структурным, то есть удаляющим целые группы весов, это позволит также ускорить проход вперед через РНН, особенно при выполнении вычилений на графических процессорах. В качестве группы весов могут выступать строки или столбцы матриц весов.

В [18, 25] было показано, что рекуррентные нейронные сети можно разреживать с помощью метода прунинга в десятки раз без потери качества решения задачи. В [17, 20, 4] были предложены эффективные методы байесовского разреживания для полносвязных сетей, в отличие от прунинга не требующие тщательной настройки гиперпараметров. Для РНН эти методы ранее исследованы не были. Целью данной работы является разработать метод байесовского разреживания РНН на основе методов [17, 20, 4] с учетом особенностей РНН.

3 Обзор научной области

3.1 Байесовские нейронные сети

В отличие от классических методов машинного обучения, в которых находят точечную оценку на параметры нейронной сети w, в байесовских нейронных сетях объекты, целевые переменные и параметры рассматриваются как случайные величины. Соответственно, нейронная сеть моделирует зависимость $p(y|x, w)$. Априорное распределение $p(w)$ задает исходные знания и ожидания о параметрах. Например, в разреживающих моделях априорное распределение поощряет нулевые значения параметров. Процесс обучения состоит в поиске апостериорного распределения на параметры $p(w|\mathcal{D})$. Тогда предсказания модели будут задаваться как

$$p(y|x) = \mathbb{E}_{p(w|\mathcal{D})} p(y|x, w).$$

Найти апостериорное распределение на параметры по формуле Байеса

$$p(w|\mathcal{D}) = \frac{p(\mathcal{D}|w)p(w)}{\int p(\mathcal{D}|w')p(w')dw'}, \quad p(\mathcal{D}|w) = \prod_{i=1}^{N} p(y_i|x_i, w)$$

не удается из-за невычисляемого интеграла в знаменателе. Поэтому ищут приближенное апостериорное распределение $q_\lambda(w)$ в некотором параметрическом семействе распределений, λ — параметры приближенного апостериорного распределения. Параметры λ выбирают так, чтобы минимизировать KL-дивергенцию:

$$KL(q_\lambda(w)||p(w|\mathcal{D})) \to \min_w,$$

что равносильно максимизации вариационной нижней оценки на логарифм правдоподобия:

$$\mathcal{L}(\lambda) = \sum_{i=1}^{N} \mathbb{E}_{q_\lambda(w)} \log p(y_i|x_i, w) - KL(q_\lambda(w)||p(w)) \to \max_\lambda \tag{1}$$

Это выражение по сути представляет собой сумму слагаемого, отвечающего за качество решения задачи, и регуляризатора, показывающего, что апостериорное распределение на параметры должно быть близко к априорному.

Первое слагаемое обычно оценивают по методу Монте-Карло с помощью одного семпла весов для каждого объекта:

$$\mathbb{E}_{q_\lambda(w)} \log p(y_i|x_i, w) \approx \log p(y_i|x_i, w), w \sim q_\lambda(w).$$

Чтобы избежать смещения градиентов, применяют трюк репараметризации [13]: задают веса в виде $w = f(\lambda, \epsilon)$, $\epsilon \sim p(\epsilon)$. Трюк репараметризации применим не ко всем распределениям, однако применим, например, к нормальному распределению:

$$w \sim \mathcal{N}(w|\theta, diag(\sigma^2)) \Leftrightarrow w = \theta + \sigma \odot \epsilon, \epsilon \sim \mathcal{N}(w|0, I),$$

$\odot$ — поэлементное умножение.

3.2 Разреживающий вариационный дропаут

Дропаут [23] — это техника регуляризации нейронных сетей, предполагающая наложение мультипликативного шума на входы каждого слоя. Обычно элементы вектора шума генерируются из распределения Бернулли (бинарный дропаут) или из нормального распределения с центром в 1 (гауссовский дропаут), и параметры этого шума настраиваются с помощью кросс-валидации. В [12] предложена интерпретация гауссовского дропаута как способа задания байесовской нейронной сети. Это позволило настраивать параметры шума автоматически. В [17] этот подход был расширен для разреживания полносвязных нейронных сетей и назван разреживающим вариационным дропаутом (РВД).

Рассмотрим полносвязный слой $h = g(Wx+b)$ с матрицей весов W. В РВД априорное распределение на веса задается в виде факторизованного лог-равномерного распределения:

$$p(W) = \prod_{i,j=1}^{k,n} p(|w_{ij}|), \quad p(|w_{ij}|) \propto \frac{1}{|w_{ij}|}$$

Это распределение имеет большую массу в нуле и поэтому поощряет нулевые веса.

Приближенное апостериорное распределение ищется в семействе факторизованных нормальных распределений:

$$q(W) = \prod_{i,j=1}^{k,n} q(w_{ij}), \quad q(w_{ij}|\theta_{ij}, \alpha_{ij}) = \mathcal{N}(\theta_{ij}, \alpha_{ij}\theta_{ij}^2)$$

Применение такого апостериорного распределения равносильно наложению мультипликативного [12]

$$w_{ij} = \theta_{ij}\xi_{ij}, \quad \xi_{ij} \sim \mathcal{N}(1, \alpha_{ij}),$$

или аддитивного [17]

$$w_{ij} = \theta_{ij} + \epsilon_{ij}, \quad \epsilon_{ij} \sim \mathcal{N}(0, \sigma_{ij}^2), \quad \alpha_{ij} = \frac{\sigma_{ij}^2}{\theta_{ij}^2}. \tag{2}$$

нормального шума на веса. Параметризация весов (2) называется аддитивной репараметризацией и позволяет уменьшить дисперсию градиентов $\mathcal{L}$ по средним весов θ_{ij}. Кроме того, поскольку сумма нормальных распределений есть нормальное распределение с вычисляемыми параметрами, шум можно накладывать на преактивацию Wx, а не отдельно на компоненты матрицы W. Этот прием называется локальной репараметризаций [12]. Локальная репараметризация позволяет еще сильнее уменьшить дисперсию градиентов, а также экономит вычисления, поскольку семплировать шум на веса отдельно для каждого объекта — это дорогая операция.

В РВД оптимизация вариационной нижней оценки (1) производится по $\{\theta, \log\sigma\}$ с применением трюка репараметризации, аддитивной репараметризации и локальной репараметризации для достижения несмещенных градиентов с низкой дисперсией. Поскольку априорное и приближенное апостериорное распределения факторизуются по весам, KL-дивергенция также распадается на сумму по отдельным весам, и каждое слагаемое зависит только от дисперсии шума α_{ij} в силу специального выбора априорного распределения [12]:

$$KL(q(w_{ij}|\theta_{ij}, \alpha_{ij})||p(w_{ij})) = k(\alpha_{ij}),$$

$$k(\alpha) \approx 0.64\sigma(1.87 + 1.49\log\alpha) - 0.5\log(1 + \alpha^{-1}) + C. \tag{3}$$

Последнее выражение является достаточно точной аппроксимацией KL-дивергенции и получено в [17].

KL-дивергенция (3) поощряет большие значения α_{ij} и маленькие по модулю значения θ_{ij}. Если $\alpha_{ij} \to \infty$ для веса w_{ij}, то из-за большой дисперсии шума модели выгодно установить $\theta_{ij} = 0$ и $\sigma_{ij} = \alpha_{ij}\theta_{ij}^2 = 0$, чтобы избежать больших ошибок предсказания. В результате распределение $q(w_{ij}|\theta_{ij}, \alpha_{ij})$ приближается к дельта-функции в 0, и данный вес всегда является нулевым.

3.3 Групповые методы байесовского разреживания нейронных сетей

В [4] модель разреживающего вариационного дропаута была расширена для достижения группового разреживания полносвязного слоя. Под групповым разреживанием понимается удаление группы весов из модели, например строки или столбцы матрицы весов. Групповое разреживание позволяет удалять элементы скрытых слоев нейронной сети, что ускоряет проход вперед. В качестве примера мы будем объединять в группы столбцы матрицы весов полносвязного слоя и нумеровать их $1 \dots k$.

Авторы [4] предлагают ввести групповые мультипликативные веса z_i для каждой группы весов и настраивать веса в следующей параметризации:

$$w_{ij} = \hat{w}_{ij} z_i.$$

В полносвязном слое эта параметризация эквивалентна наложению мультипликативного шума на вход слоя:

$$h = g(\hat{W}(x \odot z) + b).$$

Поскольку главной задачей является обнулить z_i, авторы используют для мультипликативных переменных ту же пару априорное-приближенное апостериорное распределение, что в РВД:

$$p(z_i) = \frac{1}{|z_i|}$$

$$q(z_i|\theta_i^z, \sigma_i^z) = \mathcal{N}(z_i|\theta_i^z, (\sigma_i^z)^2).$$

Для отдельных весов используют стандартное нормальное априорное распределение, а апостериорное распределение, как и в РВД, приближают в классе нормальных распределений:

$$p(w_{ij}) = \mathcal{N}(w_{ij}|0, 1)$$

$$q(w_{ij}) = \mathcal{N}(w_{ij}|\theta_{ij}, \sigma_{ij}^2).$$

Априорное распределение на отдельные веса поощряет нулевые средние θ_{ij}, а это, в свою очередь, помогает приближать групповые средние θ_i^z к нулю, то есть обнулять групповые переменные.

Обучение модели производится аналогично модели РВД с помощью оптимизации вариационной нижней оценки (1). KL-дивергенция распадается на сумму KL-дивергенций для групповых переменных и для весов, причем последнее слагаемое вычисляется аналитически.

4 Описание метода

4.1 Разреживающий вариационный дропаут для рекуррентных нейронных сетей

Для большинства задач рекуррентные нейронные сети задаются плотными матрицами весов, при этом большая часть весов может быть малоинформативной и не влиять на качество решения задачи [18]. Несмотря на существование эвристических подходов к разреживанию РНН [18, 19, 25], опирающихся на большое количество гиперпараметров, применение байесовских техник разреживания ранее не было исследовано для рекуррентных нейронных сетей. С другой стороны, в литературе описаны различные модели байесовских РНН [6, 5], некоторые особенности которых найут отражение и в предложенной модели.

При применении РВД к РНН следует учитывать особенности рекуррентного слоя:

- веса в рекуррентном слое связаны по времени, то есть разные элементы входной последовательности умножаются на одни и те же матрицы весов;
- в байесовской РНН текущее скрытое состояние h_t и матрица рекуррентных весов W^h не являются независимыми случайными величинами, поскольку второе участвовало в выражениях для вычисления первого.

Сначала рассмотрим модель разреживающего вариационного дропаута для рекуррентного слоя, а затем отметим особенности применения РВД к полносвязному слою и слою представлений в РНН.

Следуя [17], мы используем лог-равномерное априорное распределение на веса рекуррентного слоя $\{W^x, W^h\}$ и аппроксимируем апостериорное распределение в классе нормальных распределений:

$$\begin{aligned} p(w_{ik}^x) &= \frac{1}{|w_{ik}^x|}, \quad q(w_{ik}^x|\theta_{ik}^x, \sigma_{ik}^x) = \mathcal{N}\big(w_{ik}^x|\theta_{ik}^x, {\sigma_{ik}^x}^2\big), \\ p(w_{ij}^h) &= \frac{1}{|w_{ij}^h|}, \quad q(w_{ij}^h|\theta_{ij}^h, \sigma_{ij}^h) = \mathcal{N}\big(w_{ij}^h|\theta_{ij}^h, {\sigma_{ij}^h}^2\big), \end{aligned} \tag{4}$$

Обучение модели заключается в оптимизации вариационной нижней оценки

$$\begin{aligned} \sum_{i=1}^{N} \int q(w|\theta,\sigma) \log p\Big(y_i\big|f_h(x_{i,T}, f_h(\dots f_h(x_{i,1}, h_{i,0})))\Big) d\omega - \\ - \sum_{k,i=1}^{n,m} k\left(\frac{{\sigma_{ik}^x}^2}{{\theta_{ik}^x}^2}\right) - \sum_{j,i=1}^{m,m} k\left(\frac{{\sigma_{ij}^h}^2}{{\theta_{ij}^h}^2}\right) \end{aligned} \tag{5}$$

по параметрам $\{\theta, \log\sigma\}$ с помощью стохастических градиентных оптимизационных методов. В выражении (5) первое слагаемое представляет собой правдоподобие модели, усредненное по распределению на веса $q(w|\theta,\sigma)$. В процессе оптимизации это правдоподобие оценивается по методу Монте-Карло одним семплом весов. Как в модели модели РВД, здесь используются трюк репараметризации и аддитивная репараметризация с целью получить несмещенные градиенты с низкой дисперсией:

$$w_{ik}^x = \theta_{ik}^x + \epsilon_{ik}^x \sigma_{ik}^x, \quad \epsilon_{ik}^x \sim \mathcal{N}(\epsilon_{ik}^x|0,1)$$

$$w_{ij}^h = \theta_{ij}^h + \epsilon_{ij}^h \sigma_{ij}^h, \quad \epsilon_{ij}^h \sim \mathcal{N}(\epsilon_{ij}^h|0,1).$$

В правдоподобии зависимость целевой переменной y_i развернута по времени, чтобы подчеркнуть, что во все моменты времени используются одни и те же веса W^x, W^h. Таким образом, нормальный шум ϵ_{ik}^x и ϵ_{ij}^h в байесовской РНН должен быть связан по времени [6]: для одного объекта в каждый момент времени используется один и тот же семпл шума.

Однако, в байесовской РНН нельзя применить локальную репараметризацию ни для весов W^x, ни для весов W^h. Применение локальной репараметризации к матрице весов W^x в РНН подразумевает использование одного и того же семпла шума на преактивацию $W^x x_t \ \forall t$, что не равносильно использованию одного и того же семпла весов W^x во все моменты времени. Для W^h локальную репараметризацию нельзя применить еще по одной причине: поскольку h_{t-1} и W^h не являются независимыми случайными величинами, к произведению $W^h h_{t-1}$ неприменимо утверждение о сумме нормальных распределений.

Вместо использования локальной репараметризации во избежание ресурсоемкого семплирования трехмерных матриц шума предлагается использовать один семпл весов W^x и W^h для всех объектов одного мини-батча.

Аналогичную схему можно применить и для гейтовых архитектур, например, LSTM. В этом случае априорное и приближенное апостериорное распределения будут использоваться для каждой из матриц параметров.

На этапе тестирования можно использовать средние значения весов Θ^x, Θ^h по аналогии с [17, 6]. Кроме того, веса с большими значениями $\alpha_{ij} = \frac{\sigma_{ij}^2}{\theta_{ij}^2}$ обнуляются [17].

Итоговая схема одного временного шага прохода вперед по разреживающей байесовской РНН, включающая обнуление весов в тестовом режиме, приведена в алгоритме 1.

При применении разреживающего дропаута к другим слоям РНН, предшествующим или следующим за рекуррентным слоем, на этапе обучения следует, как и в рекуррент-

Алгоритм 1 Прямой проход по РНН с разреживающим вариационным дропаутом: 1 шаг по времени

Вход: входная матрица X_t — мини-батч (размера число объектов $\times$ входная размерность n);

Вход: матрица текущих скрытых состояний H_t (размера число объектов $\times$ размерность скрытого слоя m);

Вход: матрицы параметров Θ^x, Θ^h, Σ^x, Σ^h, вектор свободных членов b;

Вход: бинарный флаг: включено ли обнуление весов ;

Вход: бинарный флаг: фаза обучения или фаза теста ;

Вход: g — функция нелинейности;

Выход: матрица скрытых состояний H_{t+1} в следующий момент времени;

если фаза тестирования то

  $W^x, W^h = \Theta^x, \Theta^h$;

иначе

  сгенерировать Ξ^x — матрицу стандартного нормального шума размера Θ^x;

  сгенерировать Ξ^h —матрицу стандартного нормального шума размера Θ^h;

  $W^x = \Theta^x + \Xi^x \odot \Sigma^x$;

  $W^h = \Theta^h + \Xi^h \odot \Sigma^h$;

конец если

если обнулять веса: то

  $A^x = \log(\Sigma^x \odot \Sigma^x \oslash (W^x \odot W^x))$ // все операции покомпонентные

  $A^h = \log(\Sigma^h \odot \Sigma^h \oslash (W^h \odot W^h))$ // все операции покомпонентные

  $W^x = W^x \odot (A^x \leqslant 3)$;

  $W^h = W^h \odot (A^h \leqslant 3)$;

конец если

$H_{t+1} = g(W^x X_t + W^h H_t + b)$;

Вернуть H_{t+1}.

ном слое, использовать один и тот же семпл весов во все моменты времени для одного объекта.

Таким образом, при формулировке модели РВД для РНН учтены следующие особенности рекуррентных нейронных сетей:

1. семплирование одинакового шума на веса во все моменты времени;

2. в отличие от сетей прямого распространения, к РНН не применима локальная

репараметризация, поэтому предлагается семплировать одну матрицу весов для всех объектов мини-батча.

4.2 Групповое разреживание рекуррентных нейронных сетей

Чтобы разреживание помогало ускорить проход вперед по рекуррентной нейронной сети при выполнении вычислений на графических процессорах, нужно удалять веса группами, соответствующими одному нейрону. Для этого можно применить подход, описанный в [4]. Однако этот подход можно усовершенствовать для получения различных уровней разреженности в гейтовых рекуррентных архитектурах. Рассмотрим этот подход для наиболее популярной гейтовой архитектуры LSTM.

В LSTM [9], помимо вектора скрытого состояния, в каждый момент времени поддерживается вектор внутренней памяти c_t. На каждом временном шаге сначала обновляется память с помощью гейтового механизма, затем обновляется скрытое состояние:

$$i = \sigma(W_i^h h_{t-1} + W_i^x x_t + b_i) \quad f = \sigma(W_f^h h_{t-1} + W_f^x x_t + b_f)$$
$$g = tanh(W_g^h h_{t-1} + W_g^x x_t + b_g) \quad o = \sigma(W_o^h h_{t-1} + W_o^x x_t + b_o)$$
$$c_t = f \odot c_{t-1} + i \odot g \qquad h_t = o \odot tanh(c_t)$$

По аналогии с [4], для достижения группового разреживания введем в модель мультипликативные групповые переменные на веса. Помимо групповых переменных z^x и z^h на строки матриц весов, отвечающих за исключение элементов входного и скрытого векторов, введем также групповые переменные $z^i,\ z^f,\ z^g$ и z^o на столбцы матриц весов, отвечающие за обуславливание гейтов $i,\ f,\ o$ и информационного потока g на входные данные. Например, для матрицы W_f^x получится следующая параметризация весов:

$$w_{f,ij}^x = \hat{w}_{f,ij}^x \ \cdot \ z_i^x \ \cdot \ z_j^f.$$

Такая параметризация соответствует наложению мультипликативного шума на входной вектор x_t и скрытое состояние h_t, а также отдельных мультипликативных шумов на

преактивации гейтов и информационного потока:

$$
\begin{aligned}
i = &\ \sigma\Big(\big(\hat{W}_i^h(h_{t-1} \odot z^h) + \hat{W}_i^x(x_t \odot z^x)\big) \odot z^i + b_i\Big) \\
f = &\ \sigma\Big(\big(\hat{W}_f^h(h_{t-1} \odot z^h) + \hat{W}_f^x(x_t \odot z^x)\big) \odot z^f + b_f\Big) \\
g = &\ tanh\Big(\big(\hat{W}_g^h(h_{t-1} \odot z^h\big) + \hat{W}_g^x(x_t \odot z^x)\big) \odot z^g + b_g\Big) \\
o = &\ \sigma\Big(\big(\hat{W}_o^h(h_{t-1} \odot z^h) + \hat{W}_o^x(x_t \odot z^x)\big) \odot z^o + b_o\Big) \\
c_t &= f \odot c_{t-1} + i \odot g \qquad h_t = o \odot tanh(c_t)
\end{aligned}
$$

При обнулении компонент z^x и z^h из модели исключается элемент входного вектора или скрытого состояния соответственно. При обнулении компонент $z^i,\ z^f,\ z^g$ или z^o элемент соответствующего гейта или информационного потока становится константным, не обусловленным на входные данные x_t и h_t. Отметим, что появление в модели константных гейтов упрощает, но не нарушает структуру LSTM, и кроме того, позволяет экономить вычисления матричных произведений.

В [4] использовалось стандартное нормальное априорное распределения на отдельные веса $\hat{w}_{ij}$, однако на практике это ограничивает разреживание модели. В данной работе предлагается использовать такую пару априорное–приближенное апостериорное распределение, как в РВД для РНН, на все группы весов (для примера приведены распределения для матрицы W_f^x):

$$
\begin{aligned}
p(\hat{w}_{f,ij}^x) = \frac{1}{|\hat{w}_{f,ij}^x|}; \quad q(\hat{w}_{f,ij}^x) = \mathcal{N}(\hat{w}_{f,ij}^x | \theta_{f,ij}^x, (\sigma_{f,ij}^x)^2) \\
p(z_i^x) = \frac{1}{|z_i^x|}; \quad q(z_i^x) = \mathcal{N}(z_i^x | \theta_i^x, (\sigma_i^x)^2) \\
p(z_j^x) = \frac{1}{|z_j^x|}; \quad q(z_j^x) = \mathcal{N}(z_j^x | \theta_j^x, (\sigma_j^x)^2)
\end{aligned}
$$

За счет разреживания всех трех групп весов достигается эффект иерархии: разреживание отдельных весов способствует появлению константных гейтов и упрощению структуры LSTM, что, в свою очередь, помогает исключать элементы x_t и h_t.

Семплирование групповых переменных $z^x, z^h,\ z^i,\ z^f,\ z^g$ и z^o проводится с использованием трюка репараметризации и аддитивной репараметризации, как в модели РВД для РНН. Обучение модели, проход вперед и фаза тестирования аналогичны модели РВД для РНН с добавлением лишь дополнительного семплирования групповых переменных в проходе вперед и слагаемых KL-дивергенции, отвечающих за групповые переменные, в вариационной нижней оценке (5).

Групповое разреживание можно аналогично применить и к слою представлений. Для этого надо ввести групповые мультипликативные переменные на элементы словаря и разреживать как элементы матрицы представлений, так и мультипликативные групповые переменные. Апориорные и приближенные апостериорные представления на веса остаются такими же, как в описанной выше модели группового РВД для РНН. В результате применения такой модели достигается эффект разреживания входного словаря, то есть отбора признаков.

Таким образом, при формулировке модели группового РВД для РНН учтены следующие особенности гейтовых рекуррентных нейронных сетей:

1. введены мультипликативные переменные на преактивации гейтов и информационного потока;

2. для весов w_{ij} использовано разреживающее лог-равномерное априорное распределение, усиливающее разреживание групповых переменных.

5 Эксперименты

5.1 Описание прикладных задач и данных

Эксперименты проводились на дискриминативной задаче анализа тональности на двух наборах данных. В задаче анализа тональности для одного объекта — текста необходимо предсказать метку тональности.

Первый набор данных, Movie Reviews [21], состоит из 10 тысяч рецензий на фильм, для каждой из которых необходимо предсказать оценку фильма от 0 до 1 по десятибальной шкале. Качество решения задачи оценивается по среднеквадратичному отклонению. 20% данных были отложены в контрольную выборку.

Второй набор данных, IMDB [16], состоит из 25 тысяч обучающих и 25 тысяч контрольных объектов — рецензий на фильм. Для каждого объекта необходимо предсказать его тональность — положительную или отрицательную. Качество решения задачи оценивается по доле правильных ответов. Для целей валидации 15% обучающих данных были отложены в валидационную выборку.

5.2 Архитектура нейронной сети и ее обучение

В обеих задачах входной текст воспринимался как последовательность слов, словарь был составлен из 20 тысяч наиболее популярных слов в данном наборе данных. Для решения обеих задач использовалась рекуррентная нейронная сеть с тремя слоями:

1. слой представлений, для каждого слова возвращающий вектор представления;
2. слой LSTM;
3. полносвязный слой, по последнему скрытому представлению предсказывающий целевую переменную.

В различных экспериментах для каждого слоя либо использовался предложенный метод регуляризации — (групповой) разреживающий вариационный дропаут (РВД для РНН), либо бинарный дропаут для РНН [6] с $p = 0.3$ и коэффициентом L_2-регуляризации 10^{-3} по аналогии с экспериментами в [6]. Далее конкретный способ регуляризации будет указан в каждом эксперименте.

В задаче MovieReviews размерность представления и скрытого состояния в LSTM равнялась 128. В задаче IMDB представления инициализировались предобученными на

большом корпусе текстов векторами [22] размерности 300, размерность скрытого слоя также равнялась 128.

Обучение нейронной сети производилось с помощью стохастического оптимизационного метода Adam [11] с размером мини-батча 128, шагом 0.001 в задаче MovieReviews и 0.0005 в задаче IMDB, обучение длилось 1000 эпох.

5.3 Методы для сравнения

Сравнение проводилось с тремя методами, обеспечивающими групповую разреженность:

1. Логнормальный мультипликативный шум [20] на группы весов. В оригинальной статье метод исследован для полносвязных и сверточных сетей, однако напрямую переносится и на рекуррентные нейронные сети.

2. Байесовское сжатие [4]. Поскольку метод группового РВД для РНН получен обобщением [4], то этот метод похож на РВД для РНН с двумя отличиями: отсутствуют групповые переменные на преактивации гейтов, и для отдельных весов используется стандартное нормальное априорное распределение.

3. Структурное разреживание для LSTM [25]. Этот метод группового прунинга для LSTM предполагает группировку весов, соответствующих каждому элементу скрытого состояния, и применения $\mathrm{L}_{1/2}$ регуляризации для этих групп. Группы с маленькой L_2-нормой весов удаляются вместе с соответствующим элементов скрытого состояния LSTM. Для этого метода порог на L_2-норму группы весов и коэффициент регуляризации был подобран по валидационной выборке.

5.4 Исследование соотношения качество — разреженность в РВД для РНН

Для задачи MovieReviews было проведено исследование соотношения качества и уровня разреженности при обучении РВД для РНН из случайной инициализации и с предобучением. Для предобучения использованы две модели: нерегуляризованная LSTM с ранним остановом и предобучение с бинарным дропаутом [6]. Далее веса, полученные при предобучении, использованы для инициализации средних в РВД для РНН. В данном эксперименте для слоя представления и полносвязного слоя в качестве регуляризации используется бинарный дропаут, для слоя LSTM используется РВД. Результаты экспермента приведены в таблице 1.

Таблица 1: Качество и уровень разреженности при применении РВД для РНН с предобучением и без

Модель (иниц.)	MSE	Разреженность x - h %
LSTM без рег.	0.1518	–
LSTM с бин. дроп.	0.1488	–
РВД (случ. иниц.)	0.1526	99.91 – 99.90
РВД (предобуч. без рег.)	0.1503	99.95 – 99.92
РВД (предобуч. бин. дроп.)	0.1475	99.63 – 99.49

Эксперимент показал, что наилучшие показатели разреженности получаются при обучении модели из случайной инициализации, при этом качество решения задачи немного ухудшается относительно применения РВД для РНН с предобучением. Однако при использовании предобучения показатели разреженности получаются немного ниже. Это понятный результат: при использовании предобучения слагаемое в вариационной нижней оценке, отвечающее за качество, перевешивает разреживающий регуляризатор, и уровень разреживания получается ниже.

Тем не менее, данный эксперимент показал, что без предобучения качество решения задачи остается приемлемым, и в последующих экспериментах предобучение не выполняется.

Отметим также, что благодаря высокому уровню разреженности автоматически достигается и групповая разреженность, хотя это и не гарантируется моделью.

5.5 Исследование разреженности и интерпретируемости в групповом РВД для РНН

На данных IMDB проведен эксперимент по включению и выключению группового разреживания разных слоев РНН. Обучено три модели: с разреживанием только рекуррентного слоя (R), с разреживанием рекуррентного и полносвязного слоя (RD) и с разреживанием всех трех слоев (VRD), включая разреживание словаря. В случае, если разреживание слоя выключено, он регуляризуется бинарным дропаутом [6]. Результаты приведены в таблице 2.

Эксперимент показывает, что добиться сильного сжатия модели удается только при одновременном разреживании всех слоев. В частности, разреженность скрытого состоя-

Таблица 2: Качество и уровень разреженности при разреживании рахного подмножества слоев

Разреживаемые слои	Точность	Число оставленных		
		слов	элементов представления	скрытых нейронов
R	0.822	20000	1	18
RD	0.823	20000	1	4
VRD	0.832	268	1	5

ния LSTM получается сильно выше, если ей дополнительно способствует разреживание полносвязного слоя. При добавлении разреживания словаря удается сжать слой представлений почти в 100 раз, это сильно экономит память и время обработки объекта.

В модели с разреживанием всех слоев остается только один элемент вектора представления и 5 элементов скрытого состояния, и, кроме того, модели достаточно менее чем 300 слов для качественных предсказаний. Визуализации ниже показывают, почему этих слов достаточно.

На рис. 1 по оси абсцисс визуализирован единственный необходимый модели элемент представления для 268 оставшихся после разреживания слов. По оси ординат отложен уровень тональности слова, подсчитанный простым способом по данным: усреднением значений целевого вектора (0 или 1) для всех вхождений данного слова в тексты выборки. Визуализация показывает, что в представлении модель сохраняет для каждого слова его уровень тональности, и поскольку классификация бинарная, этого достаточно для хороших предсказаний.

На рис. 2 представлена маска константных гейтов для оставшихся 5 нейронов. Информационный поток не является константным, и это логично, иначе по нейросети перестала бы передаваться информация. Модель оставляет один полноценный элемент LSTM и 4 элемента с константными или почти константными гейтами (упрощение структуры LSTM). В итоге вместо 20 матричных произведений необходимо вычислять только 8, что экономит вычисления почти в 2.5 раза.

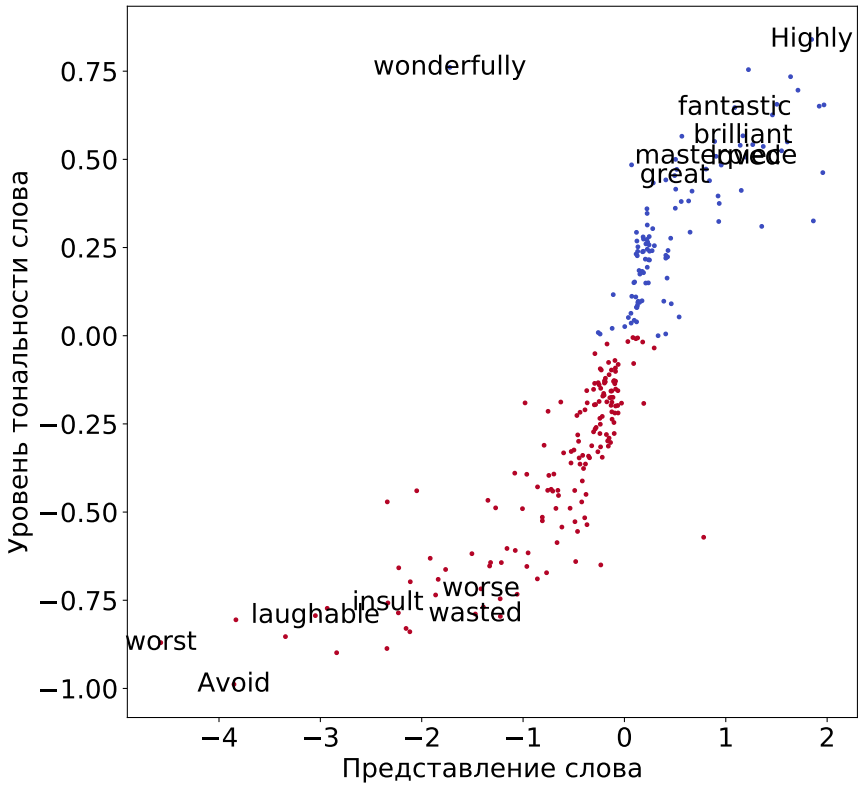

Рис. 1: Визуализация представлений оставшихся после разреживания словаря слов для задачи IMDB

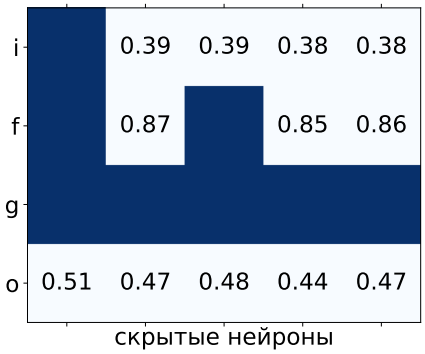

Рис. 2: Визуализация константных гейтов для задачи IMDB: голубым цветом показаны константные гейты, синим — не константные, т. е. обусловленные на данные, число в каждой ячейке показывает константное значение гейта.

5.6 Сравнение показателей разреженности и качества с другими методами

В таблице 3 приведены результаты сравнения по качеству работы и уровню разреживания РВД и группового РВД с другими методами группового разреживания РНН на задаче MovieReviews. В таблице 4 аналогичные результаты приведены для задачи IMDB. В этом эксперименте разреживание выполнялось только для рекуррентного

слоя.

Таблица 3: Сравнение РВД для РНН и группового РВД для РНН с другими методами группового разреживания РНН на данных MovieReviews

	Точность	число активных		
		вход. нейронов (из 300)	скрыт. нейронов (из 128)	гейтов
LSTM (неразреж. модель)	0.1518	–	–	–
Lognormal noise [20]	0.1539	7	65	220
Bayes. Compression [4]	0.1544	7	18	72
Sparse Structures [25]	0.1592	12	12	48
РВД	0.1519	9	22	30
Груп. РВД	0.1526	4	4	9

Таблица 4: Сравнение РВД для РНН и группового РВД для РНН с другими методами группового разреживания РНН на данных IMDB

	Точность	число активных		
		вход. нейронов (из 300)	скрыт. нейронов (из 128)	гейтов
LSTM (неразреж. модель)	0.841	–	–	–
Lognormal noise [20]	0.846	6	77	308
Bayes. Compression [4]	0.833	4	17	68
Sparse Structures [25]	0.816	9	4	16
РВД	0.82	3	8	17
Груп. РВД	0.823	1	4	8

Эксперимент показывает, что групповой РВД для РНН превосходит по уровню разреживания другие методы группового разреживания LSTM, хотя и немного уступает им в качестве. Избежать подобного эффекта можно, использовав предобучение, однако в этом случае показатели разреженности станут немного ниже.

6 Заключение

В работе показана применимость методов байесовского разреживания нейронных сетей к рекуррентным нейронным сетям, предложена модель разреживающего вариационного дропаута для рекуррентных нейронных сетей, а также модель группового разреживающего вариационного дропаута, учитывающая гейтовую структуру современных рекуррентных архитектур. В экспериментах показано, что групповое разреживание рекуррентных нейронных сетей упрощает интерпретацию модели.

Направления дальнейших исследований включают тестирование метода на более сложных задачах анализа и синтеза, например задаче распознавания и синтеза звука, а также соединение модели байесовского разреживания РНН с другими методами сжатия рекуррентных нейронных сетей: факторизацией матриц весов и представлении весов в виде чисел с меньшей точностью.

Полученные результаты могут быть применены для переноса сложных современных рекуррентных архитектур на мобильные устройства, например смартфоны. Результаты по групповому разреживанию РНН и сопутствующему ускорению обработки входного объекта могут быть применены в системах реального времени, основанных на рекуррентных архитектурах, например онлайн-переводчиках.

Список литературы

[1] Amodei, D., Ananthanarayanan, S., Anubhai, R., Bai, J., Battenberg, E., Case, C., Casper, J., Catanzaro, B., Cheng, Q., Chen, G., Chen, J., Chen, J., Chen, Z., Chrzanowski, M., Coates, A., Diamos, G., and et al. Deep speech 2 : End-to-end speech recognition in english and mandarin.

[2] Atanov, A., Ashukha, A., Molchanov, D., Neklyudov, K., and Vetrov, D. CoRR abs/1802.04893 (02 2018).

[3] Chan, W., Jaitly, N., Le, Q. V., and Vinyals, O. Listen, attend and spell: A neural network for large vocabulary conversational speech recognition. In ICASSP (2016).

[4] Christos Louizos, Karen Ullrich, M. W. Bayesian compression for deep learning. In arXiv preprint arXiv:1705.08665 (2017).

[5] Fortunato, M., Blundell, C., and Vinyals, O. Bayesian recurrent neural networks. CoRR abs/1704.02798 (2017).

[6] Gal, Y., and Ghahramani, Z. A theoretically grounded application of dropout in recurrent neural networks. In Advances in Neural Information Processing Systems 29 (NIPS) (2016).

[7] Goodfellow, I., Bengio, Y., and Courville, A. Deep Learning. The MIT Press, 2016.

[8] Ha, D., Dai, A., and Le, Q. V. Hypernetworks. In Proceedings of the International Conference on Learning Representations (ICLR) (2017).

[9] Hochreiter, S., and Schmidhuber, J. Long short-term memory. Neural Comput. 9, 8 (Nov. 1997), 1735–1780.

[10] Hubara, I., Courbariaux, M., Soudry, D., El-Yaniv, R., and Bengio, Y. Quantized neural networks: Training neural networks with low precision weights and activations. arXiv e-prints abs/1609.07061 (Sept. 2016).

[11] Kingma, D. P., and Ba, J. Adam: A method for stochastic optimization. In Proceedings of the 3rd International Conference for Learning Representations (2015).

[12] Kingma, D. P., Salimans, T., and Welling, M. Variational dropout and the local reparameterization trick. CoRR abs/1506.02557 (2015).

[13] Kingma, D. P., and Welling, M. Auto-encoding variational bayes. CoRR abs/1312.6114 (2013).

[14] Le, Q. V., Jaitly, N., and Hinton, G. E. A simple way to initialize recurrent networks of rectified linear units. CoRR abs/1504.00941 (2015).

[15] Lei Ba, J., Ryan Kiros, J., and E. Hinton, G. Layer normalization. CoRR (07 2016).

[16] Maas, A. L., Daly, R. E., Pham, P. T., Huang, D., Ng, A. Y., and Potts, C. Learning word vectors for sentiment analysis. In Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies - Volume 1 (Stroudsburg, PA, USA, 2011), HLT '11, Association for Computational Linguistics, pp. 142–150.

[17] Molchanov, D., Ashukha, A., and Vetrov, D. Variational dropout sparsifies deep neural networks. In Proceedings of the 34th International Conference on Machine Learning, ICML 2017 (2017).

[18] Narang, S., Diamos, G. F., Sengupta, S., and Elsen, E. Exploring sparsity in recurrent neural networks. CoRR abs/1704.05119 (2017).

[19] Narang, S., Undersander, E., and Diamos, G. Block-sparse recurrent neural networks, 2018.

[20] Neklyudov, K., Molchanov, D., Ashukha, A., and Vetrov, D. P. Structured bayesian pruning via log-normal multiplicative noise. In Advances in Neural Information Processing Systems 30, I. Guyon, U. V. Luxburg, S. Bengio, H. Wallach, R. Fergus, S. Vishwanathan, and R. Garnett, Eds. Curran Associates, Inc., 2017, pp. 6778–6787.

[21] Pang, B., and Lee, L. Seeing stars: Exploiting class relationships for sentiment categorization with respect to rating scales. In Association for Computational Linguistics (2005).

[22] Pascanu, R., Mikolov, T., and Bengio, Y. On the difficulty of training recurrent neural networks. In Proceedings of the 30th International Conference on International Conference on Machine Learning - Volume 28 (2013), ICML'13, JMLR.org, pp. III–1310–III–1318.

[23] Srivastava, N. Improving neural networks with dropout. PhD thesis, University of Toronto, 2013.

[24] Tjandra, A., Sakti, S., and Nakamura, S. Compressing recurrent neural network with tensor train. CoRR abs/1705.08052 (2017).

[25] Wen, W., He, Y., Rajbhandari, S., Zhang, M., Wang, W., Liu, F., Hu, B., Chen, Y., and Li, H. Learning intrinsic sparse structures within long short-term memory. In International Conference on Learning Representations (2018).

[26] Wu, Y., Schuster, M., Chen, Z., Le, Q. V., Norouzi, M., Macherey, W., Krikun, M., Cao, Y., Gao, Q., Macherey, K., Klingner, J., Shah, A., Johnson, M., Liu, X., Kaiser, L., Gouws, S., Kato, Y., Kudo, T., Kazawa, H., Stevens, K., Kurian, G., Patil, N., Wang, W., Young, C., Smith, J., Riesa, J., Rudnick, A., Vinyals, O., Corrado, G., Hughes, M., and Dean, J. Google's neural machine translation system: Bridging the gap between human and machine translation. CoRR abs/1609.08144 (2016).