Министерство образования и науки Российской Федерации
Московский физико-технический институт (государственный университет)

Факультет инноваций и высоких технологий
Кафедра анализа данных

Выпускная квалификационная работа бакалавра по направлению 01.03.02
«Прикладная математика и информатика»

Исследование методов оптимизации для обучения нейросетей на основе архитектуры Трансформер в задаче машинного перевода

Студент 694 группы
Лепехин М. Н.

Научный руководитель
Шаграев А. Г.

Долгопрудный
2020

Аннотация

Развитие технологий машинного обучения и искусственного интеллекта позволило автоматизировать решение многих прикладных задач из самых разных областей и значительно повысить качество решений. Одной из важных задач является перевод с одного языка на другой. Использование машинного перевода позволяет значительно снизить финансовые и временные затраты, позволяя переводчикам получить либо уже окончательно переведённый текст, либо достаточно качественную основу для дальнейшего перевода.

В последние годы для машинного перевода наиболее широко используются нейронные сети. При этом самые высокие результаты в машинном переводе достигаются с использованием архитектуры Трансформер. Поэтому именно перевод с помощью Трансформер рассматривается в данной работе.

Для получения качественного перевода удачного выбора модели недостаточно. Необходимо уметь правильно обучать модель. Поэтому выбор метода оптимизации, а также гиперпараметров для него имеет большое значение. Но на данный момент не существует универсального алгоритма для подбора наилучшего метода оптимизации и гиперпараметров для задаваемой языковой пары и обучающего датасета. Поэтому проблема выбора алгоритма оптимизации требует детального изучения.

В этой работе рассматриваются различные методы оптимизации первого порядка для обучения архитектуры Трансформер и исследуется их поведение в зависимости от пары языков, между которыми выполняется перевод, и размера параллельного корпуса для обучения.

Содержание

Глава 1

Введение

История нейросетевого машинного перевода

Ранее для решения задачи машинного перевода применялись иерархический и фразовый подходы. Но в последние годы они были вытеснены нейросетевым машинным переводом [1].

Развитие и внедрение нейросетей в системы машинного перевода начало происходить в 2014 году [2]. Первые успешные попытки применения нейросетевых моделей к задаче машинного перевода были основаны на использовании рекуррентных нейронных сетей. Позднее появились разнообразные техники по улучшению качества перевода. Важным прорывом в этой области стало появление механизма attention [2,3]. А в 2017 году появилась ещё более мощная модель для машинного перевода – Трансформер [4]. Эта модель не содержит ни рекуррентных, ни свёрточных слоёв, используя вместо них только полносвязные слои и механизмы attention и self-attention. Применение Трансформера позволило значительно повысить точность нейросетевого машинного перевода.

Задача нейросетевого машинного перевода

Все нейросетевые модели для машинного перевода состоят из двух частей: энкодера и декодера. Энкодер переводит токены на исходном языке в латентное векторное пространство. Декодер преобразует эти векторы в текст на целевом языке. Модель генерирует вероятностное распределение токенов на целевом языке при условии предложения на исходном языке и всех предыдущих токенов на целевом языке.

Обучение нейронной сети для машинного перевода заключается в максимизации логарифма правдоподобия референсного перевода. Его можно записать следующим образом:

$$L(\theta) = \sum_{n=1}^{N} \log p(y^{(n)}|x^{(n)};\theta) = \\ = \sum_{n=1}^{N}\sum_{t=1}^{T_n} \log p\big(y_t^{(n)}\big|y_{<t}^{(n)}, x^{(n)};\theta\big) \tag{1}$$

Если дополнительно наложить условие, что используется attention-based модель, то функция потерь преобразуется следующим образом:

$$L(\theta) = \sum_{n=1}^{N}\sum_{t=1}^{T_n} \log p\left(y_t^{(n)}\middle|y_{<t}^{(n)}, h_{t-1}^{(n)}, f^{att}\big(f^{enc}(x^{(n)}), y_{<t}^{(n)}, h_{t-1}^{(n)}\big);\theta\right) \tag{2}$$

Как правило, логарифм правдоподобия референсного предложения нормируют на длину предложения:

$$L(\theta) = \sum_{n=1}^{N}\frac{1}{T_n}\sum_{t=1}^{T_n} \log p\left(y_t^{(n)}\middle|y_{<t}^{(n)}, h_{t-1}^{(n)}, f^{att}\big(f^{enc}(x^{(n)}), y_{<t}^{(n)}, h_{t-1}^{(n)}\big);\theta\right) \tag{3}$$

Использование методов оптимизации в задачах машинного обучения

Обучение подавляющего большинства моделей в машинном обучении основано на оптимизации некоторой функции, называемой *функцией потерь*. Как правило, эта функция является непрерывной и дифференцируемой. В большинстве случаев найти точку глобального оптимума аналитически невозможно. Для решения этой проблемы используют методы оптимизации. Большая часть методов, применяемых на практике, являются модификациями градиентного спуска и называются *методами первого порядка*. К ним относятся SGD (стохастический градиентный спуск), Adagrad [10], Adadelta [11], RMSProp, Adam [12], а также разнообразные их вариации. Помимо методов первого порядка в машинном обучении иногда применяют и *методы второго порядка*, а также квазиньютоновские методы [13]. Методы второго порядка, основным из которых является метод Ньютона, имеют более высокую скорость сходимости и требуют меньше итераций до сходимости. Однако, на каждой итерации они используют матрицу, обратную гессиану оптимизируемой функции, или некоторую её оценку. Это вызывает необходимость вычислять, хранить и обращать гессиан, что при оптимизации функций от достаточно большого числа переменных это приводит к потреблению слишком большого объема памяти, а также к значительным временным затратам. Поскольку в задаче машинного перевода обычно используются сложные модели, имеющие большое число параметров, методы второго порядка не применяются на практике.

Цель работы

Для получения качественных переводов недостаточно только правильно подобранной модели. Для того, чтобы использовать весь её потенциал, нужно уметь правильно её обучать. Одной из наиболее важных вещей при обучении нейросетей является выбор алгоритма оптимизации. В данное понятие включаются как выбор непосредственного метода оптимизации, так и подбор оптимальных гиперпараметров для него. Существует работа [5], в которой в некотором виде исследуется применение методов оптимизации для обучения моделей на базе архитектуры Трансформер. Но в ней подбор оптимальных гиперпараметров рассматривается не очень подробно и авторы статьи рассматривают зависимость наилучших гиперпараметров от имеющихся вычислительных ресурсов, а не от характеристик конкретного датасета, на котором обучается Трансформер. Цель работы: проанализировать имеющиеся методы оптимизации первого порядка для обучения моделей на основе архитектуры Трансформер в задачах машинного перевода; выработать рекомендации по подбору алгоритмов и их гиперпараметров, в зависимости от объёма и типа данных. Для того, чтобы сравнивать наборы методом и гиперпараметров между собой будем смотреть на кривые на валидации целиком и делать выводы исходя из этого.

Глава 2

Обзор литературы

Архитектура Трансформер

Архитектура Трансформер, впервые опубликованная в 2017 году [4], позволила превзойти наилучший до этого BLEU на англо-немецком и англо-французском корпусах WMT 2014. Сейчас большинство улучшений качества перевода на крупных датасетах делается посредством небольших модификаций этой архитектуры [6]. Изначально предназначавшийся для задачи машинного перевода, Трансформер постепенно вытеснил рекуррентные нейронные сети в основных задачах обработки естественного языка.

Трансформер имеет ряд архитектурных преимуществ над моделями, основанных как на рекуррентных слоях, так и на свёртках. Рекуррентные нейронные сети имеют следующие проблемы: затухание градиентов, тенденцию к "забыванию"информации (для того чтобы рекуррентная сеть могла "вспомнить"информацию из начала длинного предложения в конце этого предложения, приходится значительно увеличивать размерность скрытого состояния). Кроме того, так как рекуррентные сети по определению являются последовательными, возможности для их распараллеливания ограничены. Трансформер менее склонен к "забыванию"информации, поскольку в нём связь между любыми двумя словами предложения происходит после прохождения через 1 слой. Кроме того, он также может быть легко распараллелен на уровне полносвязных слоёв и слоёв attention и self-attention. Недостатком свёрточных сетей является то, что в них связь между двумя наиболее дальними словами предложения s происходит после прохождения соответствующих эмбеддингов через $O(\log |s|)$ слоёв, что негативно складывается на способности сети связывать по смыслу слова предложения, находящиеся на большом расстоянии друг от друга.

Методы оптимизации

Рассмотрим методы оптимизации, использущиеся при обучении нейросетей для машинного перевода и решения других задач.

Обозначим оптимизируемую функцию за f.

Stochastic gradient descent

$$\theta = \theta - \eta \nabla_\theta f(\theta; x^{(i)}, y^{(i)}) \tag{4}$$

Этот метод подобно стандартному градиентному спуску использует факт, что антиградиент функции f является направлением её наибольшего убывания. Но основным отличием SGD от GD является то, что SGD за одну итерацию обновляет параметр ровно по одному элементу обучающей выборки. Это позволяет находить значение оптимальных параметров на больших датасетах значительно быстрее.

Преимущества. Одна итерация метода выполняется гораздо быстрее, чем в других методах первого порядка.

Недостатки.

1) Менее стабилен, чем стандартный градиентный спуск. Для сходимости требуется большее число итераций и, возможно, больше одного запуска.

2) При оптимизации квадратичной функции чувствителен к плохо обусловленным матрицам A.

Стоит отметить, что можно в SGD считать градиент не по одному случайно выбранному элементу обучающего датасета, а по некоторому фиксированному небольшому число элементов. Такие наборы элементов называют *мини-батчами*, а модифицированный таким образом градиентный спуск – *Mini-batch Gradient Descent*.

Momentum и Nesterov accelerated gradient Скорость сходимости GD и особенно SGD может резко падать, когда эти алгоритмы встречают на пути 'овраги' - области d-мерного пространства, в которых поверхность изгибается значительно круче в одном направлении чем в другом. Из-за этого SGD начинает осциллировать в таких областях. Цель метода momentum исправить эту ситуацию, сделав шаг алгоритма менее чувствительным к таким областям за счёт накопления шага.

$$v_t = \gamma v_{t-1} + \eta \nabla_\theta f(\theta) \tag{5}$$

$$\theta = \theta - v_t \tag{6}$$

Чем больше значение γ, тем менее метод чувствителен к резким изгибам поверхности, но тем медленнее сходимость вне таких областей. Обычно берут γ, близкое к 0.9.

Несмотря на то, что метод momentum помогает решать проблему осцилляции градиента, он имеет один недостаток. Он накапливает градиент, но на каждом шаге не учитывает значение градиента в точке, в которую он отправляется. Nesterov accelerated gradient исправляет эту проблему.

$$v_t = \gamma v_{t-1} + \eta \nabla_\theta f(\theta - \gamma v_{t-1}) \tag{7}$$

$$\theta = \theta - v_t \tag{8}$$

Adagrad Суть метода Adagrad [10] в том, что он адаптирует шаг метода под каждый конкретный параметр. Чем больше сумма квадратов частных производных по параметру $\theta^{(i)}$, тем большую частоту этот параметр имеет и с тем меньшим шагом метода его надо обновлять.

Запишем формулы для шага алгоритма. Пусть $\theta^{(i)}$ - i-я компонента вектора θ.

$$g_{t,i} = \frac{\partial f}{\partial \theta^{(i)}}(\theta_t) \tag{9}$$

$$G_{t,i} = \sum_{t' \le t} g_{t',i}^2 \tag{10}$$

$$\theta_{t+1,i} = \theta_{t,i} - \frac{\eta}{\sqrt{G_{t,i} + \varepsilon}} \cdot g_{t,i} \tag{11}$$

В матрично-векторном виде шаг алгоритма можно переписать так:

$$\theta_{t+1} = \theta_t - \frac{\eta}{\sqrt{G_t + \varepsilon}} \odot g_{t,i}. \tag{12}$$

Здесь $\odot$ обозначает произведение Адамара, т.е. поэлементное перемножение векторов.

RMSProp Алгоритм RMSProp основан на той же идее, что и алгоритм Adagrad - адаптировать learning rate отдельно для каждого параметра $\theta^{(i)}$. Однако Adagrad имеет серьёзный недостаток. Он с одинаковым весом учитывает значение квадраты градиентов как с самых первых итераций, так и с самых последних. Хотя, на самом деле, наибольшую значимость имеет модули градиентов на последних нескольких итерациях. Для этого предлагается использовать экспоненциальное сглаживание.

$$\mathbb{E}[g^2]_t = \gamma \mathbb{E}[g^2]_{t-1} + (1 - \gamma) g_t^2 \tag{13}$$

$$RMS[g]_t = \sqrt{\mathbb{E}[g^2]_t + \varepsilon} \tag{14}$$

$$\theta_{t+1} = \theta_t - \frac{\eta}{RMS[g]_t} \odot g_t \tag{15}$$

Автор алгоритма рекомендует использовать $\gamma = 0.9, \eta = 0.001$.

Adadelta Метод Adadelta [11] по формуле шага и по смыслу очень похож на RMSProp. Авторы метода заметили, что в различных методах первого порядка для оптимальной сходимости нужно брать совершенно разные значения шага метода (η), а иногда – подбирать значение η в зависимости от решаемой задачи. Чтобы избавиться от

необходимости находить наилучшее значение η. Для этого корень среднеквадратичной ошибки обновления параметра (RMS) считается теперь и для $\Delta\theta$.

$$\mathbb{E}[\Delta\theta^2]_t = \gamma\mathbb{E}[\Delta\theta^2]_{t-1} + (1-\gamma)\Delta\theta_t^2 \tag{16}$$

$$RMS[\Delta\theta]_t = \sqrt{\mathbb{E}[\Delta\theta^2]_t + \varepsilon} \tag{17}$$

$$\Delta\theta_t = -\frac{RMS[\Delta\theta]_{t-1}}{RMS[g]_t} \odot g_t \tag{18}$$

$$\theta_{t+1} = \theta_t + \Delta\theta_t \tag{19}$$

Преимущество данного метода по сравнению с RMSProp - отсутствие необходимости подбирать значения параметра η. Экспериментальным путём выяснено, что для Adadelta наилучшее значение $\gamma \sim 0.9$.

Adam Этот метод [12] совмещает в себе 2 идеи: идею метода Momentum о накапливании градиента, идею методов Adadelta и RMSProp об экспоненциальном сглаживании информации о предыдущих значениях квадратов градиентов.

Adam отличается от RMSProp только тем, что в нём дополнительно выполняется экспоненциальное сглаживание числителя. За счёт этого Adam часто работает более устойчиво в окрестности оптимального значения параметра θ^*, чем методы, использующие градиент в точке x_t, не накапливая значения градиента с прошлых шагов.

Формулы шага метода выглядят так:

$$m_t = \beta_1 m_{t-1} + (1-\beta_1) g_t, \tag{20}$$

$$v_t = \beta_2 v_{t-1} + (1-\beta_2) g_t^2, \tag{21}$$

$$\theta_{t+1} = \theta_t - \frac{\eta}{\sqrt{v_t} + \varepsilon} m_t. \tag{22}$$

Но Adam имеет существенный недостаток – при его использовании требуется очень аккуратно подбирать гиперпараметры. Кроме того, без дополнительной коррекции Adam может вообще не сходиться.

Adamax является модификацией метода Adam, в которой вместо использования l_2 нормы для градиентов используется норма l_p для произвольного $p > 1$:

$$v_t = \beta_2^p v_{t-1} + (1-\beta_2^p)|g_t|^p. \tag{23}$$

Но вычислять значение v_t при больших p неудобно – из-за погрешности вычисления ответ будет посчитан со значительной ошибкой, которая будет накапливаться от итерации к итерации. Чтобы этого избежать, используют l_∞ норму, использование которой часто даёт на практике более стабильный метод.

Вместо v_t вычисляется u_t:

$$u_t = \lim_{p \to \infty} v_t^{1/p} = max(\beta_2 v_{t-1}, |g_t|) \tag{24}$$

Существуют и другие вариации метода Adam, созданные с целью устранить его недостатки. К ним относятся Adam с техникой warm-up [9], Nadam [7], в котором совмещены идеи Adam и Nesterov Momentum. Авторы RAdam [8] стремились избавиться от необходимости проведения warm-up и настройки ряда гиперпараметров для него.

RAdam (Rectified Adam) RAdam [8] является модификацией метода Adam. Цель авторов метода – сделать его более устойчивым к изменению значений гиперпараметра и убрать необходимость дополнительно проводить warm-up и подбирать гиперпараметры ещё для него.

В RAdam меняется экспоненциальное сглаживание для v_t:

$$v_t = \frac{1}{\beta_2} v_{t-1} + (1 - \beta_2) g_t^2. \tag{25}$$

Экспоненциально затухающее среднее представляется как оценка простого затухающего среднего (simple moving average) ρ. Максимальное возможное значение simple moving average оценивается следующим образом:

$$\rho_\infty = 2/(1 - \beta_2) - 1 \tag{26}$$

Тогда на каждом шаге происходит оценка длины simple moving average:

$$\rho_t = \rho_\infty - 2t \frac{\beta_2^t}{1 - \beta_2^t} \tag{27}$$

В зависимости от значения ρ_t рассматриваются 2 случая:

1) Если дисперсия адаптивного learning rate не слишком велика, т.е. $\rho_t > 4$, то делаем её снижение:

$$l_t = \sqrt{(1 - \beta_2^t)/v_t}, \tag{28}$$

$$r_t = \sqrt{\frac{(\rho_t - 4)(\rho_t - 2)\rho_\infty}{(\rho_\infty - 4)(\rho_\infty - 2)\rho_t}}, \tag{29}$$

$$\theta_t = \theta_{t-1} - \alpha_t r_t \widehat{m}_t l_t. \tag{30}$$

2) Иначе, делаем обновление с использованием неадаптивного momentum:

$$\theta_t = \theta_{t-1} - \alpha_t \widehat{m}_t. \tag{31}$$

Именно такая регуляризация дисперсии адаптивного learning rate делает метод более устойчивым.

Lookahead Lookahead, по сути, является не метод оптимизации, а способом стабилизации работы некоторого метода оптимизации. Гиперпараметрами lookahead являются число k – период синхронизации, α – шаг lookahead, A – вспомогательный оптимизатор. При этом A принимает на вход f – функцию потерь, x – текущее значение параметра θ и d – батч, на котором нужно посчитать градиент и сделать один шаг вспомогательного оптимизатора, получив x' – новое значение параметра θ.

Шаг Lookahead выглядит следующим образом.

На каждой итерации метода происходит k итераций синхронизации. На каждой итерации синхронизации происходит генерация случайного batch d из данных, на котором делается один шаг алгоритма A, в результате чего кратковременное значение параметра обновляется следующим образом:

$$\widehat{\theta}_{t,i} = \widehat{\theta}_{t,i-1} + A(f, \theta_{t,i-1}, d), i = 1, \ldots, k. \tag{32}$$

После того, как k итераций синхронизации завершились, происходит обновление долговременного параметра.

$$\theta_t = \theta_{t-1} + \alpha(\widehat{\theta}_{t,k} - \theta_{t-1}) \tag{33}$$

Если рассмотренные прежде методы оптимизации первого порядка используют небольшой шаг, потому что иначе метод будет сильно осциллировать рядом с точками локальных оптимумов, то согласно оригинальной статье [15], Lookahead даёт лучшую сходимость при использовании большого learning rate во вспомогательном методе A.

За счёт взятия большого шага в методе A будет увеличиваться сходимость метода Lookahead. А за счёт итераций синхронизации эта сходимость будет устойчивой в точках сильного искривления пространства.

Ranger Метод Ranger, появившийся в 2019 году, – Lookahead над методом RAdam. По задумке авторов, Ranger должен совмещать преимущества RAdam (отсутствие необходимости подбирать большое количество параметров) и Lookahead (большую устойчивость метода).

Связанные статьи

Отметим, что ранее были работы, в которых для задачи машинного перевода анализировались различные методы оптимизации и их гиперпараметры. Например, в [5] приведён ряд рекомендаций по подбору различных настроек эксперимента для обучения Трансформер. Но в той работе рассматриваются только 2 метода – Adam и Adafactor – и исследуется зависимость оптимальных гиперпараметров от доступных вычислительных ресурсов и размера батча. В нашей же работе упор делается на анализ методов оптимизации при фиксированных ресурсах и размере батча.

Возможна ситуация, когда публикуется метод и показываются его преимущества,

но через некоторое время выходит другая статья, которая показывает, что часто эти преимущества не проявляются. В оригинальной статье к RAdam было показано его превосходство над Adam: он давал результаты, сравнимые с Adam+warm-up, без использования warm-up и подбора большого числа гиперпараметров. Однако, в том же 2019 году вышла статья [16], показывающая, что часто RAdam может так же сильно осциллировать, как и Adam без warm-up.

Поскольку во всех адаптивных методах первого порядка нужно перебирать больше параметров чем в SGD и выполнение одной итерации таких методов медленнее, важно понять, насколько лучше они работают по сравнению с SGD. Для моделей с механизмом внимания есть результаты, показывающие, что адаптивные методы работают лучше и почему так происходит. В 2019 году была опубликована статья [14], объясняющая на примере дообучения BERT [17] для задачи классификации, что качество результатов с использованием Adam напрямую связано с распределением шума стохастических градиентов: если распределение шума имеет "тяжёлые хвосты то Adam достигает лучших результатов, иначе – стоит использовать SGD. По мнению авторов, для attention-based моделей характерно наличие "тяжёлых хвостов"в распределении шума, что приводит к низкой эффективности SGD по сравнению с Adam.

Глава 3

Эксперименты

Датасеты Были проведены эксперименты на двух наборах датасетов, содержащих пары предложений для обучения (train) и валидации (valid). Первая группа датасетов получена из параллельного корпуса TED Talks PT-EN. В полном датасете около 100000 предложений, в датасете для валидации - 3000. Для анализа зависимости методов оптимизации от размера датасета были получены 2 вспомогательных поддатасета – из 50000 и 25000 предложений.

Для анализа зависимости методов оптимизации от языковой пары были рассмотрены публичные датасеты, содержащие предложения из протоколов заседания ООН [1]. Для перевода были рассмотрены 3 направления: французско-английское, французско-испанское и русско-английское.

Параметры экспериментов Для предварительной токенизации всех рассматриваемых датасетов использовался Moses Decoder, а для получения токенов BPE — Subword NMT. Число токенов BPE бралось равным 20000. В качестве модели перевода использовался Трансформер конфигурации small (2 слоя в декодере и энкодере, 4 головы attention, размер скрытого состояния равен 256, размер скрытого состояния feed-forward сети равен 512). При обучении главная отслеживаемая метрика – BLEU на валидационном датасете. Для анализа была выбрана именно этот критерий качества, так как BLEU – автоматически вычисляемая метрика, относительно адекватно отражающая обобщающую способность модели для перевода. а

Каждый эксперимента были проведен три раза с различными значениями $random\_seed$. Для каждого эксперимента строился доверительной интервал для значения BLEU на каждой итерации. В центре каждого доверительного интервала – выборочное среднее по 3 экземплярам экспериментам, ширина доверительного интервала равна удвоенной коррекции Бесселя с одной степенью свободы.

Adam

Как уже было сказано выше, нет большого смысла в том, чтобы использовать Adam без warm-up. Для стандартизации процесса обучения использовалось расписа-

[1] opus.nlpl.eu

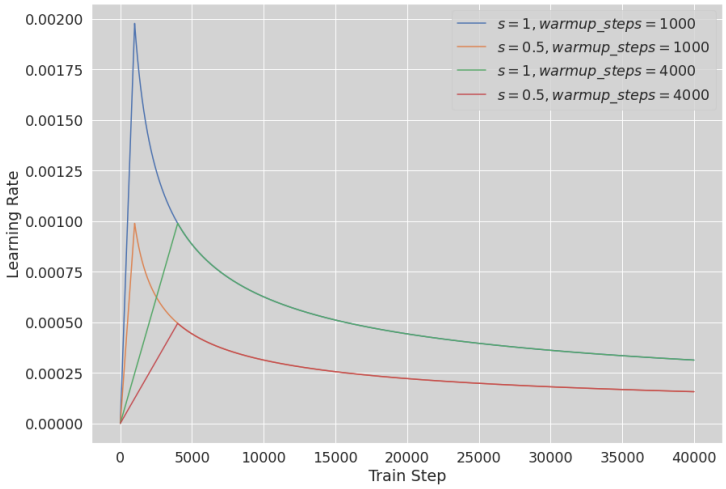

Рис. 1. Adam warmup

ние warm-up из статьи про Transformer [4]. Это расписание можно записать следующей формулой:

$$lr_{warmup\_steps}(step) = \sqrt{d_{model}} \cdot \min(\sqrt{step}, step \cdot warmup\_steps^{-3/2}). \tag{34}$$

Для Adam можно перебирать следующие гиперпараметры:

1) β_1 – параметр экспоненциального сглаживания для momentum (стандартное значение = 0.9),

2) β_2 – параметр экспоненциального сглаживания для квадрата градиента (стандартное значение=0.999),

3) гиперпараметры warm-up:

3.1) s – параметр масштаба, т.е. коэффициент, на который домножалось всё расписание warm-up,

3.2) $warmup\_steps$ – количество батчей до достижения пикового значения адаптивного learning rate.

С учётом параметра s формула изменяется следующим образом:

$$\widehat{lr}_{s,warmup\_steps}(step) = s \cdot \sqrt{d_{model}} \cdot \min(\sqrt{step}, step \cdot warmup\_steps^{-3/2}) \tag{35}$$

Графики learning rate с использованием данной стратегии warm-up для нескольких наборов гиперпараметров изображены на рис. 1.

RAdam

Для RAdam можно перебирать следующие гиперпараметры:

1) β_1 – параметр экспоненциального сглаживания для momentum (стандартное значение = 0.9),

2) β_2 – параметр экспоненциального сглаживания для квадрата градиента (стандартное значение=0.999),

3) learning rate.

Для параметров β_1 и β_2 были взяты стандартные значения по аналогии с Adam.

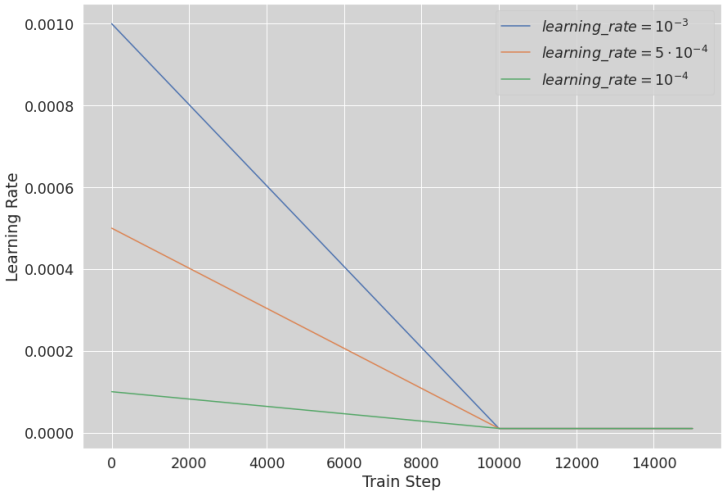

Рис. 2. RAdam learning rate

Для метода RAdam используется не константное расписание learning rate, а стратегия (рис. 2), предложенная в оригинальной статье по методу.

Ranger

Для Ranger можно перебирать следующие гиперпараметры:

1) гиперпараметры внутреннего метода (RAdam),

2) α – внешний learning rate (стандартное значение $= 0.5$),

3) k – количество итераций применения внутреннего метода (стандартное значение $= 6$).

Зависимость оптимальных гиперпараметров от размера датасета

Для исследования этой зависимости был необходим набор датасетов, отличающиеся своим размером, но все остальные характеристики (языковая пара, домен, распределение длин) которых одинаковы. Поэтому для выполнения экспериментов был использован исходный датасет TED Talks PT-EN, описанный выше, а также его сэмплированные подмножества из 25000 и 50000 предложений. Если на исходном датасете, состоящем из приблизительно 100000 предложений, обучение происходило в течение 50 эпох, то для датасетов из 50000 и 25000 предложений Трансформер обучался в течение 100 и 200 эпох соответственно. Остальные гиперпараметры эксперимента – те же, что и для исходного датасета. При этом для объективного сравнения качества модели, обученных на разных по размеру долях датасета, валидационная и тестовая выборки берутся те же, что и в экспериментах с полным датасетом.

При обучении использовался batch_size $= 256$. Для оптимизации функции потерь применялся метод Adam с warmup и со стандартными значениями гиперпараметров $\beta_1 = 0.5$, $\beta_2 = 0.999$. При обучении BLEU на датасете для валидации считается раз в каждые 1000 батчей. При переводе максимальная длина переведённого предложения

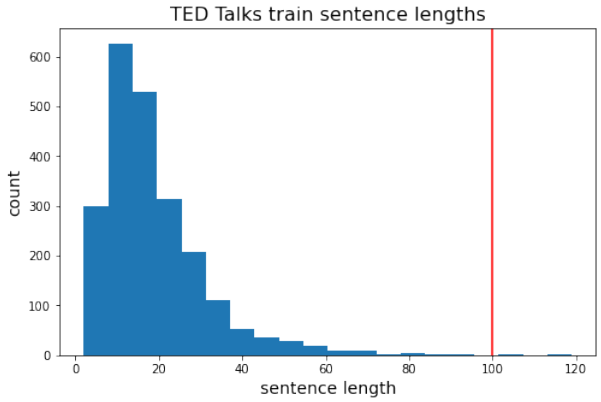

Рис. 3. Распределение длин англоязычных предложений TED Talks PT-EN

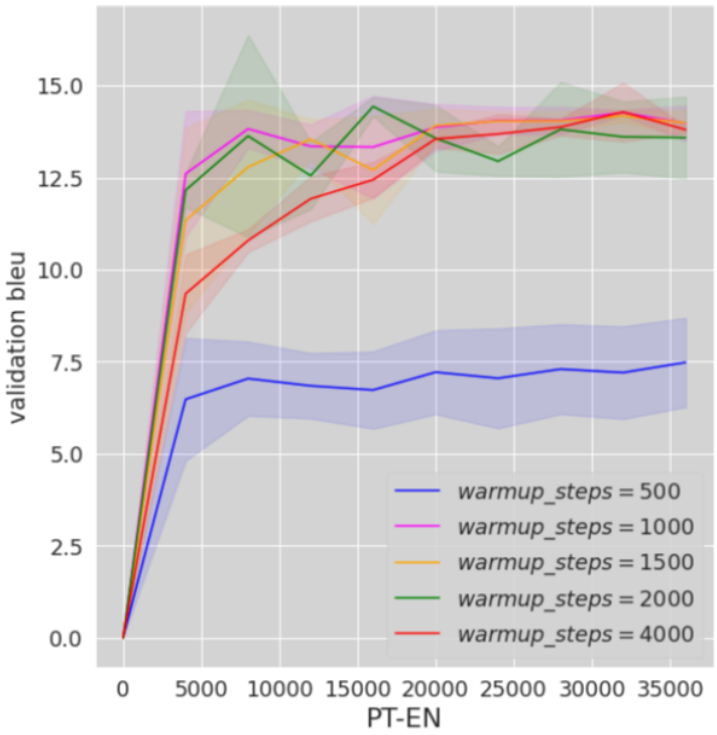

Рис. 4. Выбор *warmup_steps* для Adam

в токенах ограничивалась числом 100. Такое значение максимальной длины было выбрано по рапределению длин англоязычных предложений (рис. 3).

Adam

Были рассмотрены различные пары значений (β_1, β_2) в окрестностях стандартного значения. Было получено, что отклонение от стандартных значений не приводит к повышению BLEU на валидационной выборке. Отсутствие статзначимого улучшения видно по табл. 1. Поэтому в дальнейших экспериментах брались стандартные значения β_1 и β_2.

Следующий перебираемый параметр для Adam – *warmup_steps*. На рис. 4 видно, что при взятии *warmup_steps* меньше 1000 BLEU значительно ухудшается, но при *warmup_steps* $>$ 1000 улучшения практически не видно. Поэтому в дальнейшем все эксперименты с Adam провдятся с *warmup_steps* $= 1000$

Таблица 1. Наилучшее BLEU при выборе *warmup_steps* для Adam

Значение *warmup_steps*	BLEU
500	7.48 ± 1.21
1000	14.26 ± 0.12
1500	14.18 ± 0.16
2000	14.43 ± 0.25
4000	14.28 ± 0.8

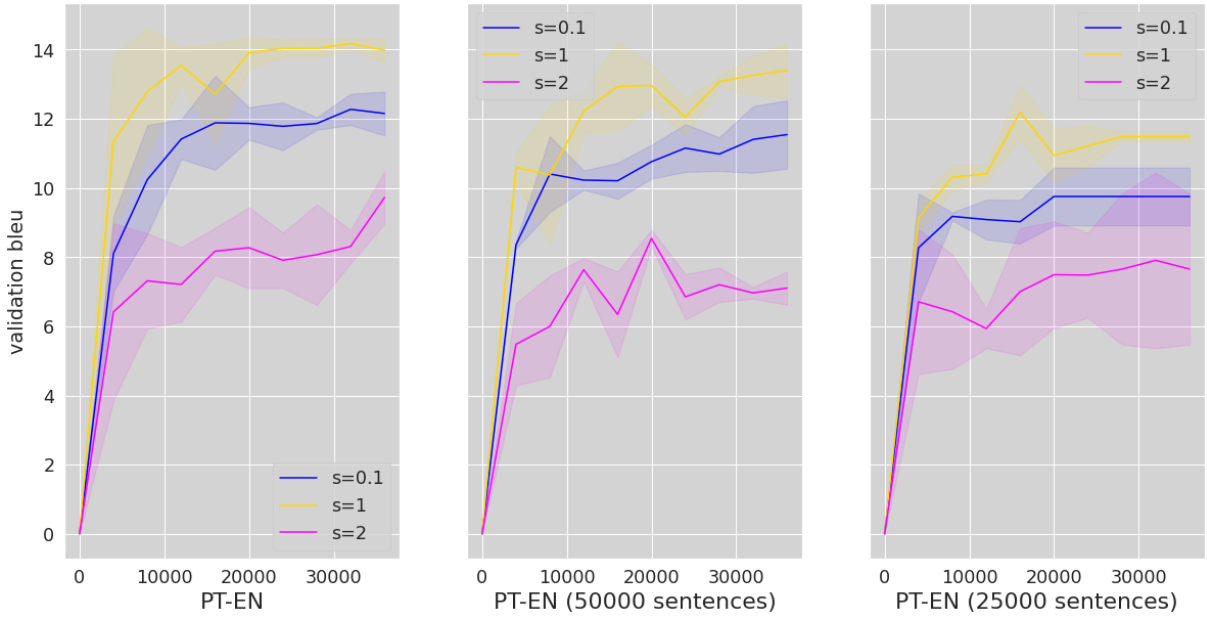

Рис. 5. Выбор s для Adam

Последний перебираемый гиперпараметр для Adam – s (рис. 5). Стандартное значение ($s = 1$) стабильно даёт высокое качество для любого размера датасета. Видно, что с уменьшением размера датасета прирост BLEU от увеличения s становится больше. Однако доверительный интервал при больших значениях s становится значительно шире, что говорит о неустойчивости метода при таком выборе s.

Ranger

Сначала перебирался learning rate внутреннего метода. Для всех остальных гиперпараметров брались значения по-умолчанию. На рис. 6 можно увидеть, что Ranger весьма устойчив к смене внутреннего learning rate. При этом, для любого размера датасета использование стандартного значения ($lr = 0.001$) подходит лучше всего. Для дальнейших экспериментов с Ranger на TED Talks PT-EN использовался learning rate=0.001.

Согласно рис. 7, оптимальное значение α для Ranger не зависит от языковой пары. Кроме того, по табл. 2 видно, что при понижении α в 5 раз (с 0.5 до 0.1) значение BLEU не снижается сильно. Это говорит об устойчивости метода к отклонению параметра α от нормального значения.

Сравнение методов

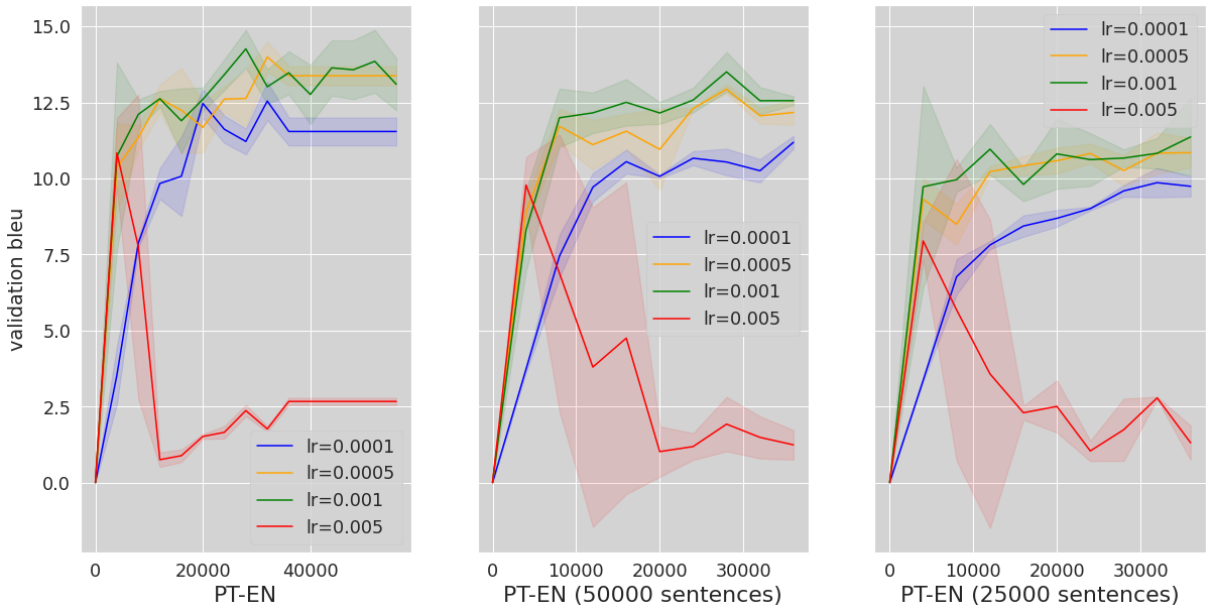

Рис. 6. Результаты работы Ranger на TED Talks PT-EN

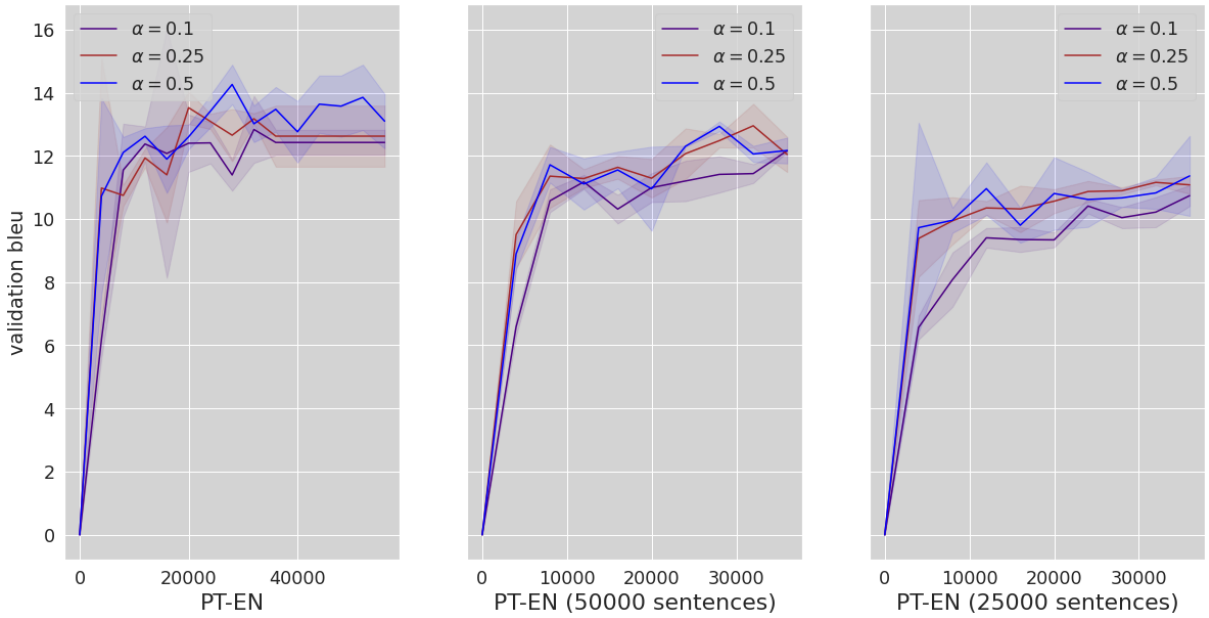

Рис. 7. Результаты работы Ranger на TED Talks PT-EN

Таблица 2. Наилучшее BLEU при выборе α для Ranger

Датасет	Значение α	BLEU
PT-EN full	0.1	12.84 ± 1.06
PT-EN full	0.25	13.52 ± 0.52
PT-EN full	0.5	14.26 ± 0.63
PT-EN 50k sentences	0.1	12.16 ± 0.1
PT-EN 50k sentences	0.25	12.95 ± 0.7
PT-EN 50k sentences	0.5	12.93 ± 0.15
PT-EN 25k sentences	0.1	10.74 ± 0.32
PT-EN 25k sentences	0.25	11.16 ± 0.1
PT-EN 25k sentences	0.5	11.36 ± 1.27

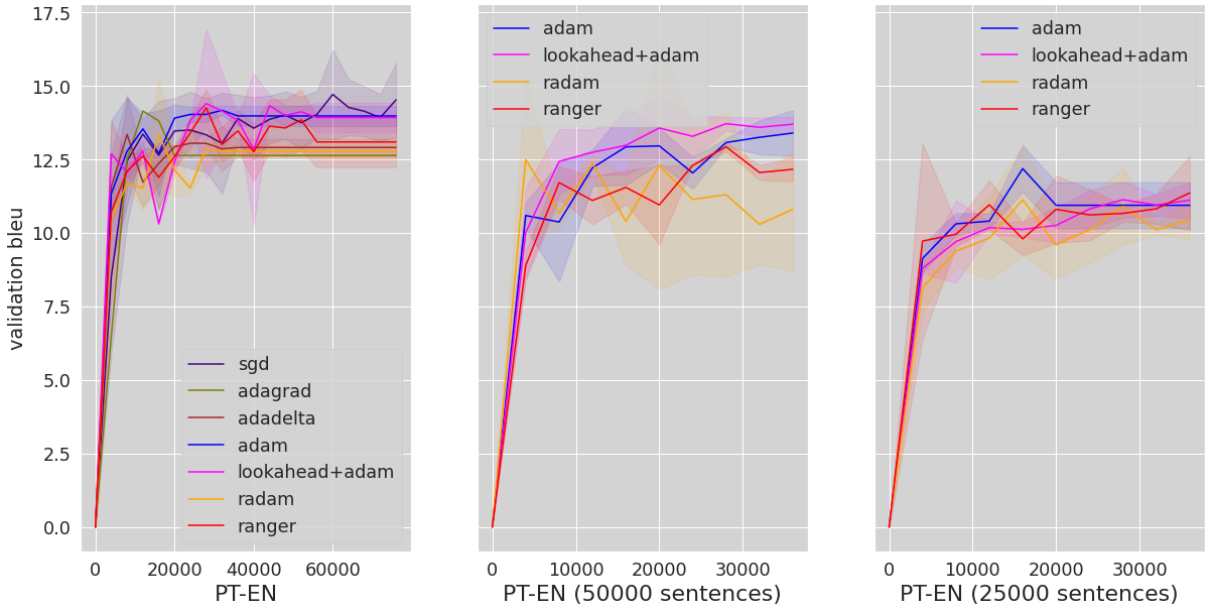

Рис. 8. Все методы на TED Talks PT-EN

Общий рис. 8 показывает, что не существует единого метода, превосходящего остальные сразу на всех размерах датасета – на полном датасете в итоге получается примерно одинаковое BLEU для Adam, Lookahead+Adam и SGD, на наборе из 50000 предложений наилучший результат достигается с использованием Lookahead+Adam, на самом маленьком датасете Ranger работает не хуже всех остальных. Тем не менее, можно заметить, что Lookahead+Adam на всех датасетах работает не хуже чем Adam, а Ranger – не хуже чем RAdam. Отсюда можно сделать вывод, что применение Lookahead к базовому методу действительно улучшает его сходимость. Кроме того, заметно, что при использовании Lookahead доверительные интервалы получаются более узкими, что говорит о более высокой устойчивости методов с Lookahead.

В самом конце были на полном TED Talks PT-EN рассмотрены методы SGD, Adagrad и Adadelta, с которыми не предполагалось проводить эксперименты изначально из-за предположения, что они будут заметно уступать Adam и другим со-

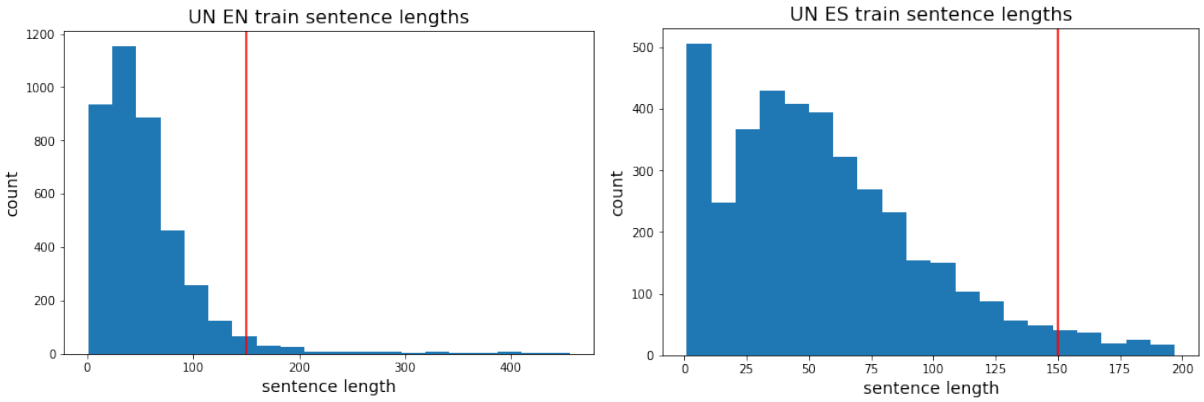

Рис. 9. Распределение длин в датасетах ООН

временным методам. В частности, для SGD это предположение было основано на результатах статьи [14]. В результате выяснилось, что эти методы при тщательном подборе гиперпараметров дают весьма неплохой результат, хотя Adagrad и Adadelta значительно уступают Adam. SGD в конце даёт одно из наиболее высоких значений BLEU. Однако, для сходимости и выхода на плато ему нужно больше итераций обучения чем Adam.

Зависимость оптимальных гиперпараметров от языковой пары

Для объективного исследования зависимости поведения методов от выбранной языковой пары, необходимо подобрать обучающие датасеты так, чтобы все остальные характеристики в них (размер датасета, качество данных, распределение длин предложений, домен) были приблизительно одинаковыми. Поэтому для экспериментов были выбраны тексты протоколов заседаний ООН. Среди языковых пар рассматриваются перевод с французского на английский, с французского на испанский и с русского на английский. Соответствующие корпуса для каждого языка содержат переводы одного и того же множества предложений на данный язык. Поэтому размер датасета, домен текстов и распределение длин примерно совпадают. Поскольку эти документы – официальные, то предполагается, что данные в них достаточно чистые.

В целом параметры эксперимента совпадают с использующимися в эксперименте на полном англо-португальском датасете. Однако, в связи с тем, что в корпусах ООН гораздо больше предложений, число токенов в которых превышает 100 (распределение длин изображено на рис. 9), длина перевода здесь ограничена 200 токенами.

Adam

По рис. 10 можно заметить, что оптимальным параметром масштаба warm-up является $s = 0.5$. Можно предположить, что такое отличие по сравнению с результатами на TED Talks PT-EN связано с большим разбросом длин предложений в

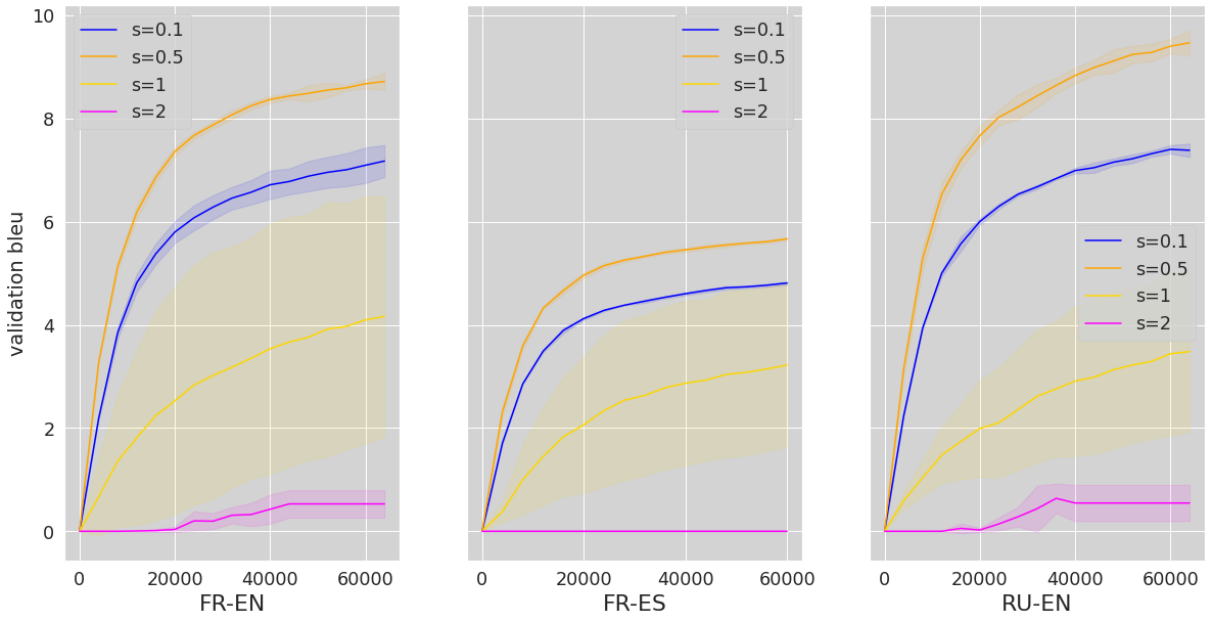

Рис. 10. Результаты работы Adam на датасетах OOH

датасетах OOH. При этом видно, что поведение метода при каждом фиксированном значении s не зависит от выбора языковой пары. Также можно отметить, что получаемое значение BLEU на французско-испанском датасете значительно ниже чем для других языковых пар. Это можно объяснить большим разбросом длин предложений на испанском языке.

RAdam

Рис. 11 показывает, что оптимальное значение learning rate лежит между 0.0005 и 0.001 для всех языковых пар. В целом, ситуация для RAdam аналогична той, что наблюдалась для Adam. Языковая пара не влияет на область оптимальных гиперпараметров. Возникает гипотеза, что главным фактором, влияющим на качество перевода, является разброс длин предложений на целевом языке. К сожалению, нам не удалось проверить эту гипотезу из-за отсутствия необходимых вычислительных ресурсов.

Ranger

Для экспериментов с Ranger для всех гиперпараметров внутреннего метода брались стандартные значения ($\beta_1 = 0.9, \beta_2 = 0.999$, learning rate=0.001). Рис. 12 для Ranger похож на то, что мы наблюдали для RAdam. Для Ranger оптимальное значение α не зависит от языковой пары. Кроме того, видно, что при понижении α в 5 раз (с 0.5 до 0.1) значение BLEU не снижается сильно. Это говорит об устойчивости метода к отклонению параметра α от нормального значения.

Сравнение методов

Подытожим результаты. По рис. 13 заметно, что на датасетах OOH поведение методов отличается от того, что мы наблюдали на TED Talks PT-EN. Видно, что Adam статзначимо превосходит другие методы – доверительные интервалы не пересекаются. Также видно, что для близких друг к другу языковых пар (французский-

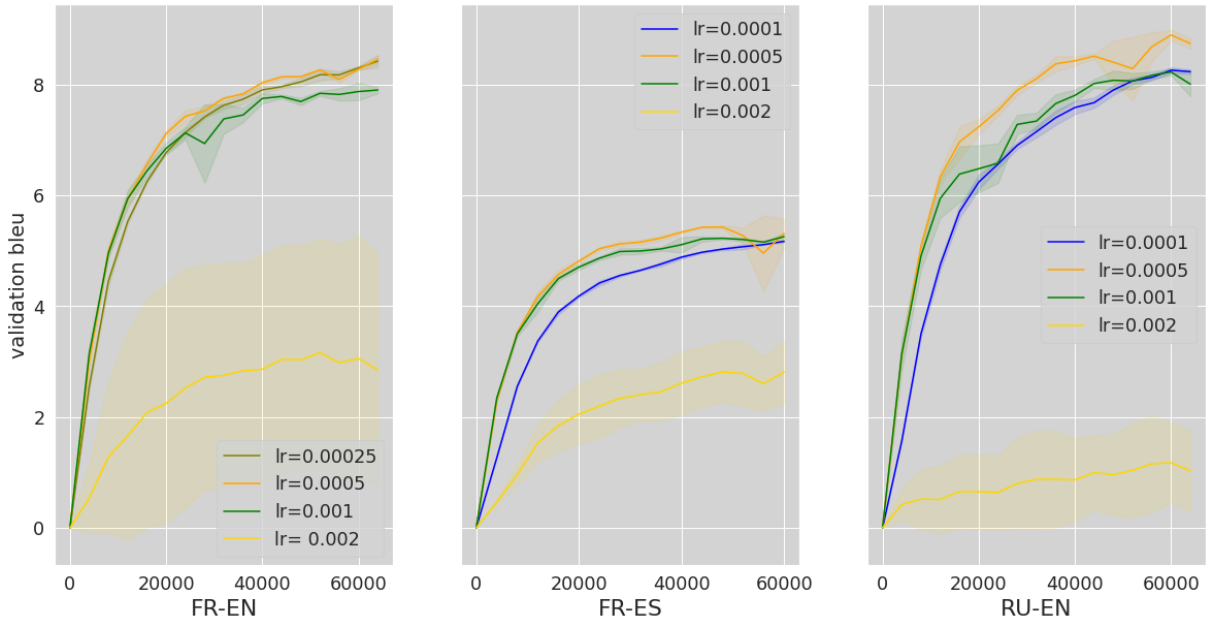

Рис. 11. Результаты работы RAdam на датасетах OOH

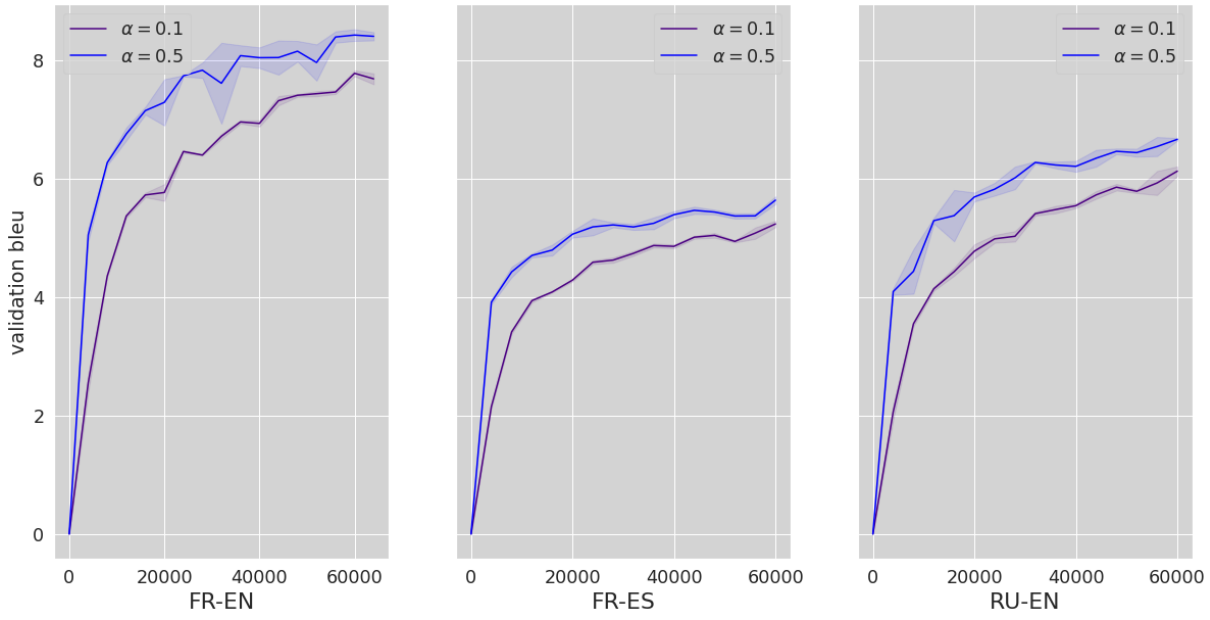

Рис. 12. Результаты работы Ranger на датасетах OOH

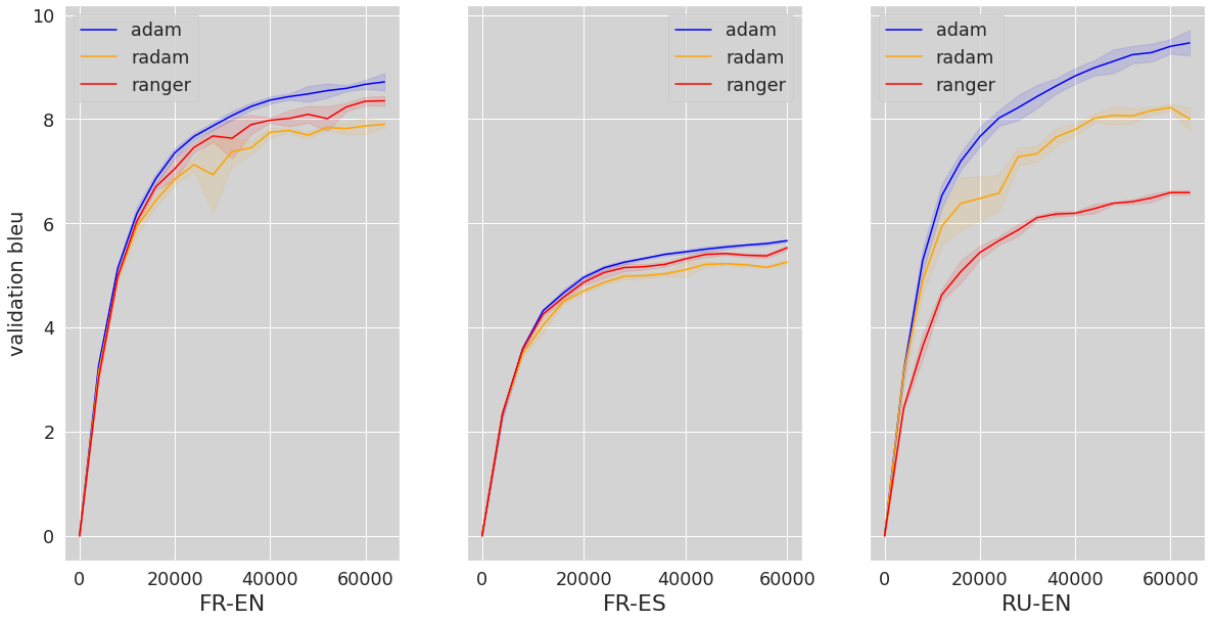

Рис. 13. Все методы на датасетах OOH

испанский и французский-английский) Ranger работает лучше чем RAdam, для более далёких языков ситуация противоположна. При этом для любой языковой пары с некоторой эпохи доверительный интервал для Ranger становится более узким чем доверительный интервал для RAdam. Это говорит о большей устойчивости Ranger.

Глава 4

Заключение

Были получены следующие итоговые выводы:

1. В большинстве случаев Adam с warm-up с оптимальными параметрами работает лучше чем RAdam. Это особенно заметно для языков, наиболее сильно отличающихся друг от друга (русский и английский). Но при этом значение s нужно тщательно подбирать, так как, например, на датасетах OOH использование стандартных значений параметра приводит к гораздо худшему результату чем RAdam. Из этого следует, что для получения относительно высоких результатов в случае отсутствия возможности подбирать гиперпараметры стоит использовать RAdam со стандартными настройками. Если же даже незначительная потеря качества критична, то нужно аккуратно перебрать гиперпараметры Adam.

2. SGD позволяет добиваться весьма высокого BLEU, но для его сходимости требуется большее число итераций чем для других методов.

3. На датасетах OOH оптимальный learning rate практически не зависит от языковой пары.

4. Использование Lookahead всегда делает метод более устойчивым, а для относительно близких друг к другу языков ещё и значительно повышает BLEU на валидации. Однако, стоит помнить, что использование Lookahead замедляет итерацию метода, что может быть критично для обучения тяжеловесных конфигураций Трансформера. Поэтому если даже небольшое ухудшение качества является критичным, то стоит использовать Lookahead; иначе – Adam или RAdam.

5. Оптимальные значения α для Lookahead-like алгоритмов не зависят от размера датасета и языковой пары.

В результате работы:

1. Написана библиотека [1] на Tensorflow 2.0 + Keras для проведения экспериментов

[1] https://gitlab.com/MikeLepekhin/methods-of-optimization-in-machine-translation

над архитектурой Трансформер с любыми методами оптимизации;

2. Проведено исследование зависимости поведения методов оптимизации от размера датасета и от выбора языковой пары;

3. Был выработан ряд общих рекомендаций по подбору алгоритма оптимизации.

Литература

[1] H. Setiawan et al., “Phrase-Based Statistical Machine Translation:A Level of Detail Approach”. *Natural Language Processing – IJCNLP 2005*

[2] D. Bahdanau, K. Cho, Y. Bengio, “Neural Machine Translation by Jointly Learning to Align and Translate”. arXiv:1409.0473

[3] M. Luong, H. Pham, C. D. Manning, “Effective Approaches to Attention-based Neural Machine Translation”. arXiv:1508.04025

[4] A. Vaswani, “Attention Is All You Need”. arXiv:1706.03762

[5] M. Popel, O. Bojar, “Training Tips for the Transformer Model”. arXiv:1804.00247

[6] E. Bugliarello, N. Okazaki, “Improving Neural Machine Translation with Parent-Scaled Self-Attention”. arXiv:1909.03149

[7] T. Dozat, “Incorporating nesterov momentum into adam”. 2016.

[8] L. Liu et al., “On the Variance of the Adaptive Learning Rate and Beyond”. arXiv:1908.03265

[9] A. Gotmare et al., “A Closer Look at Deep Learning Heuristics: Learning rate restarts, Warmup and Distillation”. arXiv:1810.13243

[10] J.Duchi, E.Hazan, Y.Singer, “Adaptive Subgradient Methods forOnline Learning and Stochastic Optimization”. Journal of Machine Learning Research 12 (2011) 2121-2159

[11] M.D. Zeiler, “ADADELTA: An Adaptive Learning Rate Method”. arXiv:1212.5701

[12] D. P. Kingma, J. Ba, “Adam: A Method for Stochastic Optimization”. arXiv:1412.6980

[13] J. Dennis, J. Moré. Quasi-Newton Methods, Motivation and Theory. SIAM Review, Societyfor Industrial and Applied Mathematics, 1977, 19 (1), pp.46-89. 10.1137/1019005. hal-01495720

[14] J. Zhang et al., Why ADAM Beats SGD for Attention Models, arXiv:1912.03194

[15] M. R. Zhang et al., Lookahead Optimizer: k steps forward, 1 step back, arXiv:1907.08610

[16] J. Ma, D. Yarats, On the adequacy of untuned warmup for adaptive optimization, arXiv:1910.04209

[17] J. Devlin et al., BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding, arXiv:1810.04805