Параллельная реализация обратного распространения нейронной сети Программное обеспечение на многопроцессорных компьютерах

Виктор Г. Царегородцев

Институт вычислительного моделирования СО РАН, Красноярск, Россия
tsar@neuropro.ru

Абстрактный. Эксперименты нейронной сети процедуры обучения распараллеливания проводятся. Несколько стилей распараллеливаний описываются и сравниваются оценки нейронного размера сети и обучение заданного размера, которые позволяют форсировку по двум-процессорам SMP машины получаются.

1. Введение

Искусственный нейронная сеть является гибок инструментом для решения многих проблем, включая нелинейную регрессию, охраняемое обучения и распознавание образов, не- контролируемого обучения, ассоциативная память, оптимизацию задачи и т.д. Здесь мы исследуем только обратное распространение нейронной сети, введенную в 1986 году и названы так из По- сколько основной части ее алгоритм обучения делает «обратное распространение ошибок» для вычисления вектора градиента вдоль регулируемого (обучаемый) переменных.

Эксперименты работы нейронных сетей на параллельных компьютерах начинаются в конце 1980-х годов [1,2,3], но в основном сосредоточены на Спецификой архитектуры - транс-счетными машинами, подключение машин, ЭВМ с массовым параллелизмом. Эта эпоха детального его исследования концов в середине 1990-х годов с замечательными работами [4,5,6,7,8] (смотри также ссылки в них). Кроме того, результаты работы [8] были подтвер- rmed недавно [9,10,11]: так называемые онлайн-обучение теоретически быстрее, чем периодического обучения, но unparallelizable (пакетное обучение, которое аккумулирует штрафы и обновления за закономерностями тренировочного набора может быть распараллеливание, но в целом сходится медленно).

Здесь изучаются распараллеливание для SMP (симметричных многопроцессорных систем) вычислимости ров. Это исследование становится необходимыми из-за широкое использование многопроцессорных серверов, технологией HyperThreading, которая была введена корпорацией Intel в последнее время, и планах корпорации Intel на этап до многоядерных процессоров, каждое ядро которого будет поддерживать HyperThreading (т.е. может работать два потока одновременно под некоторые ограничения). Таким образом, в ближайшее время мы сможем запускать до 4 параллельных потоков на одном двухъядерный процессор и SMP вычисления будут обычного использования.

Такая перспектива эволюции аппаратных средств делает SMP-программирование более ценным, чем кластеризация методов из-за широко использование общих ком- счетных машин (с multikernel процессоров в них) в ближайшее время. Кроме того, результаты, полученные для SMP-программного обеспечения дают некоторые ориентиры для кластера платформ тоже.

В статье мы кратко описать нейронную структуру сети и обучение gorithm и ал возможные схемы распараллеливания, затем описывают базы данных, используемые в

Эта работа была поддержана региональным Научным фондом Красноярска, грант 15G277.

В. Малышкин (ред.):. PaCT 2005, LNCS 3606, стр 186-192, 2005 гр

эксперименты и обеспечить экспериментальные результаты. Затем мы обсудим некоторые дополнительные вопросы и перспективы.

2 искусственная Neural Networks

2.1 Структура нейронной сети

Здесь мы используем только наиболее удобную структуру сети - с прогнозированием сети с одним скрытым слоем нейронов. Для входного вектора Икс из *N* компоненты и вектор желаемых результатов Y из *м* Компоненты можно описать нейронную сеть следующим образом. Во-первых, мы вычислим выходы скрытого слоя *N* нейроны как

$Z_я = e\left(\sum_{к=1}^{N} Икс_к вес_{мн} + вес_{0я}\right)$ для каждого, что нейрон *я* с помощью нелинейной функции *е*, обычно

сигмоидальной форма а-ля $F(\theta) = \theta / (c + |\theta|)$, $C > 0$. Тогда мы вычисляем каждый *j*- й выход

сигнал $Y_J = \sum_{л=1}^{N} Z_L U_{LJ} + U_{0к}$ переменные $вес_{об}$, $U_{компонент\ диск}$ являются обучаемые сети коэф фициенты FFI

которые должны быть скорректированы в процессе обучения. Обучение должно свести к минимуму ди далее erences между желаемым и полученными сигналами (Y_J и Y_J соответственно), используя некоторые ошибки измерить, например, из средней квадратичной формы ошибки.

Ди Ф.Ф. метода оптимизации различна может быть использована в процессе обучения - случайный поиск, генетические алгоритмы, градиентные методы оптимизации. Здесь мы используем последний быть- причины «обратное распространение ошибки» (точнее, обратное распространение частичного изводных эту меры ошибок) алгоритм позволяет быстро вычислению вектора градиента функции ошибки измерения по значениям обучаемых сетевой переменных. Когда мы получим вектор градиента, мы можем использовать градиентное уравнение спуска, чтобы улучшить качество ответа сети, сделав шаг по направлению антиградиента.

2.2 Схема и возможности распараллеливания обучения

Здесь мы используем пакетное обучение - накопление градиентов для всех моделей, собранными в обучающем наборе, чтобы вычислить общий градиент: для измерения ошибки $ЧАС_я(Икс_я, Y_я) = ||Y_я - \hat{}Y_я(Икс_я)||$ для учебного шаблона $\{Икс_я, Y_я\}$ и общая ошибка $H = \Sigma\ ЧАС_я$ мы можем вычислить $\nabla ЧАС$ в качестве $\nabla H = \nabla \Sigma\ ЧАС_я = \Sigma\ \nabla ЧАС_я$, Поэтому вычисление ди Ф.Ф. различны $ЧАС_я$ «S и их градиенты могут выполняться параллельно над ди и след различны части обучающего набора. Т.е. для двух параллельных потоков и *K* модели обучения можно разделить обучающий набор на наборы с индексами модели {1, ..., *K*/2} и { *K*/2 + 1, ..., *K*}, Одновременно вычисление $ЧАС_1$ и $\nabla ЧАС_1$ на первый и $ЧАС_2$ и $\nabla ЧАС_2$ с помощью второго потока, а затем получить фи NAL значения $H = H_1 + ЧАС_2$ и $\nabla H = \nabla ЧАС_1 + \nabla ЧАС_2$.

Противно, онлайн-обучение сети поправляет после каждой обработки рисунка (по $\nabla ЧАС_я$, не $\nabla ЧАС$) - это unparallelizable из-за частые Модите фи катионы сети и потерю пригодности для любого другого параллельно вычисленного градиента (который становится устаревшим после коррекции модели вдоль параллельной одной).

Каждая партия-тренинг эпоха состоит из общего вычисления градиента, unnec- обходимо фазы размера шага и / или выбор направления спуска (например, с использованием конъюгата

Градиент метод) и сетевое видоизменение. Итерации последние до требуемого значения *ЧАС* получаются или несколько критериев остановки выполнены, например, локальные минимумы найдены.

Для SMP-распараллеливания, т.е. быстрого доступа к памяти без каких-либо медленных сетевых соединений, мы предлагаем следующие три схемы разделения данных:

1. просит темы для следующего необработанного рисунка, т.е. нет жесткого или формального разделения обучающего набора. Но мы должны дополнительно синхронизировать доступ и Модите катион фи указателя на следующий шаблон.
2. Жесткий разделение - обучающее множество делится на части, число которых равно числу процессоров без redivisioning в последнее время. Каждый процессор (нить) здесь будет работать с постоянным подмножеством шаблонов, которые могут быть в кэше.
3. Жесткого разделения, но если поток фи искль чительна его предписанная обработка данных, он обрабатывает некоторые в настоящее время необработанных данных, назначенных на другой поток. Третья схема необходима, потому что мы можем остановить $я$-й учебный образец, когда желаемое $ЧАС_я$ получается. Таким образом, мы можем пропустить V $ЧАС_я$ вычисление, и когда количество уже подготовленных образцы в наборах соответствовало ди Ф.Ф. различны нити ди далее ERS сильно, некоторые потоки могут фи Ниш свою работу раньше и должны ждать других.

3 Базы данных, используемые в экспериментах

Мы используем 20 реальных баз данных, доступных из http://kdd.ics.uci.edu/. Все они классифицируются задачи фи катиона. Таблица 1 баз данных реферирования свойство (количество классов, количество обучающих примеров и количество байт для хранения препро cessed базы данных и некоторой дополнительной информации, необходимой) и нейронные сети свойства (число нейронов, входные сигналы, число адаптивных переменных в сети). Для три баз данных мы используем нейронные сети два ди Ф.Ф. различны размера для того, чтобы распространить результаты дальше по шкале размеров сети.

4 Результаты экспериментов

свойства распараллеливания была реализована в авторской собственной нейронной сети бег пакета программ под MS Windows. Мы не использовали пакет распараллеливания высокого уровня, но создавать и синхронизации потоков с помощью Window API. Нейронные сети ядро ранее было запрограммировано бережными достаточно - без ориентации объекта, который может повредить производительности, с ручным перепрограммированием некоторых подпрограмм, использующих Ассемблер.

Эксперименты проводились на станции SMP с двумя процессорами Pentium III 1 ГГц. В течение последних двух схем распараллеливания (как пронумеровано в разделе 2.2) мы также изучаем реализацию с жесткой уступкой каждого потока в де бесконечном процессор для того, чтобы максимально увеличить кэш удары для не изменяющих разделов обучающего набора. Любопытно, что эта версия поможет увеличить убыстрение дальше (около $3 \div 5\%$) не для небольших баз данных, но и для великих тоже, где можно кэшировать только небольшое количество данных, но это результаты не показаны здесь.

Результаты экспериментов представлены на фигурах 1-3, где вертикальные оси подсчета ускорение полученных по начальной однопоточных версии.

Таблица 1. База данных размеров и соответствует нейронным размерам сети

База данных Num. из Num. шаблоны	из Num. входных сигналов	из Num. классы	из Num. нейронов	адаптивного размера данных, имя переменные	в байтах
AnnThyroid 3772	21	3	10	253	437552
Машина 1728	6	4	15	169	110592
Hypothyroid 3162	19	2	10	222	316200
Письмо 20000	16	26	25	1101	5600000
грибы 8124	111	2	10	1142	3802032
Мускус 6598	166	2	10/15	1692/2537	4539424
питомник 12960	8	5	20	285	1036800
OptDigits 3823	62	10	10	740	1284528
PageBlocks 5473	10	5	10	165	481624
PenDigits 7494	16	10	10	280	1139088
спутниковое 4435	36	6	20	+866	887000
челнок 43500	9	7	15	262	4350000
Spambase 4601	57	2	25	1502	1159452
Гласный 990	11	11	15	356	138600
дрожжи 1484	8	10	40	770	178080
MF-Фак 2000	216	10	10/15	2280/3415	1904000
MF-Fou 2000	76	10	15	1315	784000
MF-Кар 2000	64	10	10	760	688000
MF-Pix 2000	240	10	10/15	2520/3775	2096000
MF-Зер 2000	47	10	15	880	552000

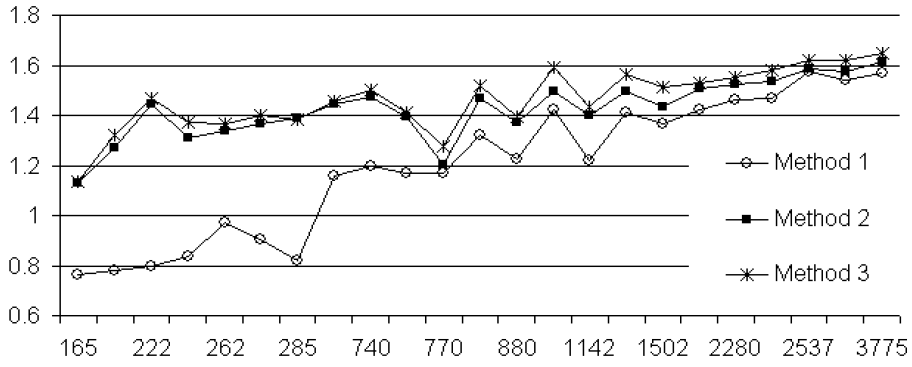

Рисунок 1. Форсировочная получены с использованием двух потоков по одной нити. Метки на горизонтальной оси, от простого упорядочения нейронных сетей по их размеру

Рис.1 показывает SpeedUp значения, когда нейронные сети просто упорядочены по их размеру. Е Ф.Ф. ectivenesses соответствует схемам распараллеливания нумерации, что является достаточно ясно: для схемы, где потоки одновременно задавать на следующую незамедленной обработку шаблона достаточно велика долю времени выбрасываются из син- хронизации ждет. Когда набор данных разделяются между потоками третьей схемой с адаптивным захватом некоторых примеров НЕОБРАБОТАННОЙ другими пробегами резьбы быстрее, чем схемы с не-помогать к каждому-другим потокам.

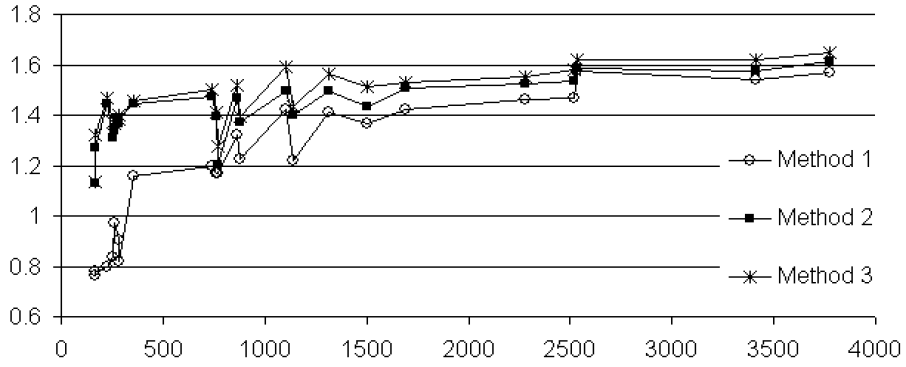

Рис. 2. Те же самые результаты, как на фиг.1, но рассчитывает горизонтальной оси в режиме реального размера сети

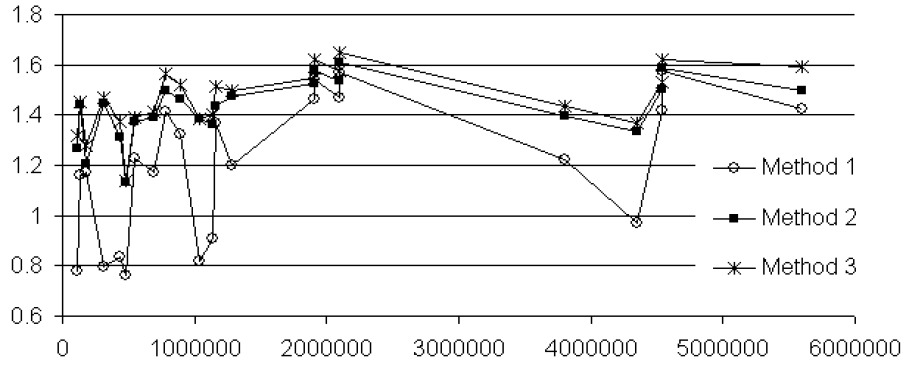

Рис. 3. Зависимости между размером данных, в байтах, и достигается ускорение

Рис.2 показывает те же результаты, рис.1, но горизонтальная ось соответствует реальному числу адаптивных параметров в сети, так что мы можем увидеть более ясно, что одновременно (первую) одна схемы не дает ускорения для сетей с менее чем 300 COей ffi- циенты. Оба Рис.1 и Рис.2 показывают улучшение с -го ускорения роста размера сети, так что можно сделать вывод, что параллелизм может быть е FFI Cient (дает, по крайней мере, убыстрение 1,5 для двух процессоров компьютера) только для обратного распространения сетей с 1000 синапсов, по крайней мере. Сеть такого размера подходит для большого числа реальных проблем, и это ди FFI культ фи-й задачей, где требуется сеть с 10000 или более весом - но только для таких крупных сетей мы можем добиться ускорения вблизи теоретического предела-
,

Следует отметить, что вычисления во время обучения нейронной сети, в основном, инструкций умножения / накопления (см раздел 2), что может быть е FFI ciently кодируются компилятором, используя инструкции SSE (или программист сам по себе может использовать пакет вычисления вектор-матрицы с SSE оптимизации в ней) - это является причиной для велика в фло uence потоков синхронизации процедур, поскольку каждую эпоху обучения для малых и средних сетей последних только доли секунды.

Там может существовать некоторую зависимость между отношением и ускорения базы данных размером (например, размером кэша в фле uence), но менее определенным. База данных размером ТФ ЭСТ на е и далее ective- Несс меньше размера сети, все убыстрение падает на Рис.3 соответствуют небольших сетей.

5. Вывод

Изучается е и далее ectiveness нейронной сети программного обеспечения распараллеливания на двух- процессоров SMP компьютера, изучить три метода разделения данных и некоторые дополнительные трюки. Полученные результаты являются многообещающими: по большей электронной FFI фективной схемы мы получаем среднее ускорение около 1,5 в течение однопоточных программы, а также увеличение скорости варьируется от 1,2 до более чем 1,6 в широких парах сети и данных размеров.

Мы планируем шаг к гетерогенным параллельным scemes, которая необходима для гигиенического perThreading особенности процессора Intel Pentium IV Prescott, где два потока может работать одновременно только при использовании ди-FF различен блоков CPU, например фли с плавающей запятой и общие блоки назначения. Кроме того, мы будем более тщательно изучить методы и в фл uental вещей, для других сетевых структур и методов тоже [12].

Ссылки

1. Beyon T. Параллельная реализация алгоритма обратного распространения по сети транспьютеров / Proc. Первый IEEE Int. Neural Network Conf. 1987.
2. Мурали П., Вечслер, Х., Манохар, М. отказоутойчивости и изучение эффективности алгоритма обратного распространения с помощью массово-параллельной реализации / Proc. из Frontiers'90, Третий симпозиум по границам Massively Parallel Processing. College Park, штат Мэриленд, США. 1990. - pp.364-367.
3. Paugam-Moisy H. На параллельный алгоритм обратного распространения путем разбиения обучающего набора / Proc. Пятый Int. Conf. Нейронные сети и их применение. Ниме, Франция. 1992 - pp.53-65.
4. Кумар В., Шекхар S, Amin MB масштабируемого параллельная разработка алгоритма обратного распространения для гиперкубов и связанных с ними архитектур / IEEE Trans. на параллельных и распределенных систем, 1994. Vol.5. Выпуск 10. - pp.1073-1090.
5. локальность и нагрузки Prechelt L. Данные балансировки для параллельного обучения нейронной сети / Proc. Семинар по Составителям для параллельных компьютеров. Испания, 1995. - pp.111-127.
6. Мишр М. Параллельные среды для реализации нейронных сетей / Нейронного Computing Surveys, 1997. Т.1. - pp.48-60.
7. Стрей А. EpsiloNN - инструмент для абстрактного Специфического катиона и параллельного моделирования нейронных сетей / Системный анализ - моделирование - моделирование, специальный выпуск по моделированию искусственных нейронных сетей, 1999. Vol.34. № 4.
8. Torresen Дж, Томита С., Landsverk О. Отношение частоты обновления веса для сходимости BP / Proc. World Conf. Нейронные сети (WCNN'1995). Вашингтон, США. 1995. - pp.679-682.
9. Уилсон DR, Мартинес TR Общее ине FFI фективности периодического обучения для градиентного спуска обучения / нейронных сетей. 2003, Vol.16. Выпуск 10. - pp.1429-1451.
10. Bottou Л., LeCun Y. Крупномасштабное онлайн обучение / Прогресс в области нейронных систем обработки информации 16 (2003). MIT Press, 2004. - pp.217-224.

11. Царегородцев В.Г. Генеральной ини FFI фективность использования градиента пакетного для нейронной сети обучения / Proc. XII конф. «Нейроинформатика и их приложения». Красноярск, Россия. 2004. - pp.145-151. (на русском).
12. Царегородцев В.Г. Перспектива распараллеливания обработки данных нейронной сети и анализ программного обеспечения / Proc. III конф. «Математика, информатика, управление». Иркутск, Россия. 2004. - 6p. (на русском).