Сети Хопфилда

Сергей Николенко

Академический Университет, весенний семестр 2011

Outline

Как работает мозг

- Как работает наша память? Мы запоминаем ассоциации. Например, надеюсь, «16 : 00 в среду» — «лекция по machine learning».
- Потом нам говорят — «16 : 00 в вреду» или (что главное) «вторая половина дня в среду», а мы припоминаем — там же лекция будет.

Как работает компьютер

- Как работает память компьютера? Компьютер запоминает массивы данных.
- Можно, конечно, использовать избыточное кодирование и защититься от небольшого количества ошибок.
- Но это не настоящая ассоциативность. Как добиться того, чтобы по размыто–ошибочному образу появлялась нужная ассоциация?

Зачем это надо

- Зачем нужна ассоциативная память?
- Первый пример — распознавание образов. Чем разные картинки похожи друг на друга? Как по искажённой картинке получить ассоциацию на её значение?

Обучение по Хеббу

- Обучение по Хеббу (Hebbian learning) — это математическая реализация ассоциативной памяти.
- Пусть есть нейронная сеть, в которой каждый нейрон x_i отвечает за какое-то событие.
- При этом каждый нейрон связан с каждым, и веса у них изменяются в соответствии с корреляцией между событиями:

$$\frac{dw_{ij}}{dt} \approx \mathrm{Corr}(x_i, x_j).$$

Обучение по Хеббу

- Теперь это работает так: каждый раз, когда в $16:00$ в среду происходит лекция, вес между этими событиями увеличивается.
- Поэтому потом, на стадии применения сети, когда сеть «вспоминает» одно из этих событий, она с высокой вероятностью ассоциирует его с другим.
- Это обучение не требует учителей, тестовых примеров с готовыми ответами (unsupervised learning) — учится просто из происходящего.

Сети Хопфилда

- Сети Хопфилда нужны как раз для того, чтобы научить компьютер ассоциативно мыслить.
- Как вы уже догадались, сеть Хопфилда — это нейронная сеть, представляющая собой полный граф.
- Нейроны — линейные с лимитом активации; для нейрона x_i:

$$a_i = \sum_j w_{ij} x_j, \qquad x_i(a_i) = \begin{cases} 1, & a \geq 0 \\ -1, & a < 0. \end{cases}$$

Синхронные и асинхронные обновления

- Важный момент: поскольку сеть с обратной связью (feedback), надо понять, синхронно или асинхронно мы проводим апдейты весов.
- Синхронно — это когда все веса считают свой результат одновременно и одновременно меняются.
- Асинхронно — когда по одному.

Суть метода

- Суть в том, чтобы сеть Хопфилда сходилась к заранее заданному набору *воспоминаний* $\{x^{(i)}\}_i$.
- Тогда, с чего бы мы ни начали, мы придём к одному из имеющихся воспоминаний, то есть вызовем самую близкую ассоциацию.
- Воспоминание — это множество значений каждого веса $\{x_j^{(i)}\}_j$.

Обучение сети Хопфилда

- Если мы хотим запомнить набор $\{x^{(i)}\}_i$, то весам присваиваем, по методу Хебба, значения, связанные с корреляциями:
$$w_{ij} = \eta \sum_k x_i^{(k)} x_j^{(k)}.$$
- Здесь η никакой роли не играет, можно, например, сделать η обратной числу воспоминаний, чтобы веса не росли слишком.

Непрерывные сети Хопфилда

- То были дискретные сети. Бывают и непрерывные, где нейроны работают по tanh:

$$a_i = \sum_j w_{ij} x_j, \qquad x_i = \tanh(a_i).$$

- Тут уже значение η имеет значение; или можно его фиксировать, а вместо этого ввести другой гиперпараметр

$$x_i = \tanh(\beta a_i).$$

О сходимости

- Мы бы хотели, чтобы сети сходились куда нам надо.
- Для этого неплохо было бы, чтобы они вообще сходились.
- Давайте попробуем доказать, что непрерывная сеть Хопфилда при известном правиле пересчёта весов действительно сходится.

Outline

Двунаправленная ассоциативная память

- Начнём со знакомого аппарата: нейронных сетей.
- Рассмотрим нейронную сеть с двумя слоями: входным и выходным.
- Входной слой получает вход, пересчитывает свои результаты и передаёт их выходному слою.

Двунаправленная ассоциативная память

- Новизна в том, что теперь второй слой, пересчитав свои результаты, отдаёт их обратно входному слою.
- И процесс итеративно продолжается.
- Идея в том, чтобы сеть достигла какого-то равновесия, стабильного состояния.
- Такие сети называются *резонансными*, или *двунаправленной ассоциативной памятью* (BAM).

Двунаправленная ассоциативная память

- Если в первом слое n перцептронов, во втором k, то получается матрица весов $\mathbf{W}$ размером $n \times k$.
- На вход поступает вектор $\mathbf{x}_0$ (строка), который преобразуется в вектор $\mathbf{y}_0$.
- Мы будем использовать линейные перцептроны с лимитом активации:

$$\mathbf{y}_0 = \mathbf{sgn}(\mathbf{x}_0\mathbf{W}).$$

Двунаправленная ассоциативная память

- Потом $\mathbf{y}_0$ подают на вход; новый шаг происходит как

$$\mathbf{x}_1^\top = \operatorname{sgn}(\mathbf{W}\mathbf{y}_0^\top)$$

(получаем из вектора длины k вектор длины n).

- И так далее; получается последовательность пар $(\mathbf{x}_i, \mathbf{y}_i)$:

$$\mathbf{y}_i = \operatorname{sgn}(\mathbf{x}_i\mathbf{W}), \quad \mathbf{x}_{i+1}^\top = \operatorname{sgn}(\mathbf{W}\mathbf{y}_i^\top).$$

Двунаправленная ассоциативная память

- Вопрос: сойдётся ли процесс? То есть дойдём ли мы до векторов $\mathbf{x}$ и $\mathbf{y}$:

$$\mathbf{y} = \operatorname{sgn}(\mathbf{x}\mathbf{W}), \quad \mathbf{x}^\top = \operatorname{sgn}(\mathbf{W}\mathbf{y}^\top).$$

- Если да, получится ассоциативная память: мы дали один вектор, а потом после нескольких итераций сеть «вспомнила» дополнительный к нему вектор, и наоборот.
- Более того, сеть вспомнила бы ассоциацию, даже если бы вектор был немножко не такой, как раньше — всё сошлось бы к ближайшей паре $(\mathbf{x}, \mathbf{y})$.

Двунаправленная ассоциативная память

- Чтобы обучить BAM, можно использовать хеббовское обучение.
- Когда мы хотим запомнить всего одну ассоциацию, матрица корреляций между двумя векторами — это просто $\mathbf{W} = \mathbf{x}^\top \mathbf{y}$. Тогда

$$\mathbf{y} = \operatorname{sgn}(\mathbf{x}\mathbf{W}) = \operatorname{sgn}(\mathbf{x}\mathbf{x}^\top \mathbf{y}) = \operatorname{sgn}(\|\mathbf{x}\|^2 \mathbf{y}) = \mathbf{y},$$

$$\mathbf{x}^\top = \operatorname{sgn}(\mathbf{W}\mathbf{y}^\top) = \operatorname{sgn}(\mathbf{x}^\top \mathbf{y}\mathbf{y}^\top) = \operatorname{sgn}(\mathbf{x}^\top \|\mathbf{y}\|^2) = \mathbf{x}^\top.$$

Двунаправленная ассоциативная память

- Но можно хранить и несколько ассоциаций $(\mathbf{x}_1, \mathbf{y}_1), \ldots, (\mathbf{x}_m, \mathbf{y}_m)$:

$$\mathbf{W} = \mathbf{x}_1^\top \mathbf{y}_1 + \ldots + \mathbf{x}_m^\top \mathbf{y}_m.$$

- Для этого случая будет лучше, если векторы $\mathbf{x}_i$ и $\mathbf{y}_i$ будут между собой попарно ортогональны.

BAM и её функция энергии

- Рассмотрим BAM со стабильным состоянием $(\mathbf{x}, \mathbf{y})$. Мы сейчас в положении $(\mathbf{x}_0, \mathbf{y}_0)$.
- Определим вектор возбуждений (excitation vector):

$$\mathbf{e}^\top = \mathbf{W}\mathbf{y}_0.$$

- Получается, что система в стабильном состоянии, если $\mathbf{sgn}(\mathbf{e}) = \mathbf{x}_0$.
- То есть если вектор $\mathbf{e}$ достаточно близок к $\mathbf{x}_0$.

BAM и её функция энергии

- Значит, можно ввести энергию

$$E = -\mathbf{x}_0\mathbf{e}^\top = -\mathbf{x}_0\mathbf{W}\mathbf{y}_0^\top,$$

  и она будет тем меньше, чем ближе $\mathbf{e}$ к $\mathbf{x}_0$.
- E получается мерой того, насколько мы близки к стабильному состоянию.

BAM и её функция энергии

- Если обобщить это просто на BAM с матрицей $\mathbf{W}$, то на шаге $(\mathbf{x}_i, \mathbf{y}_i)$ функция энергии определяется как

$$E(\mathbf{x}_i, \mathbf{y}_i) = -\frac{1}{2}\mathbf{x}_i \mathbf{W} \mathbf{y}_i^\top.$$

- $\frac{1}{2}$ пригодится позже, просто для удобства.
- Теперь мы можем доказать, что BAM рано или поздно сойдётся к стабильному состоянию.

BAM и её функция энергии

- Заметим, что $E(\mathbf{x}, \mathbf{y})$ можно переписать в двух разных видах:

$$E(\mathbf{x}, \mathbf{y}) = -\frac{1}{2}\sum_{i=1}^{k} e_i y_i = -\frac{1}{2}\sum_{i=1}^{n} g_i x_i,$$

  где $\mathbf{e} = \mathbf{x}\mathbf{W}$ — возбуждения нейронов второго слоя, а $\mathbf{g} = \mathbf{W}\mathbf{y}^\top$ — первого слоя.
- Будем рассматривать асинхронные апдейты: во время t мы случайно выбираем, какой перцептрон пересчитывать.

BAM и её функция энергии

- Состояние i-го перцептрона первого слоя изменится, только если g_i и x_i не совпадают в знаке.
- И в таком случае x_i заменится на $x_i' = \mathbf{sgn}(g_i)$.
- Поскольку остальные при этом асинхронном апдейте не меняются, энергия изменяется как

$$E(\mathbf{x}, \mathbf{y}) - E(\mathbf{x}', \mathbf{y}) = -\frac{1}{2} g_i (x_i - x_i') > 0.$$

- Значит, энергия уменьшается на каждом шаге, а всего комбинаций возможных состояний конечное число.

Вариационные методы

- Теперь мы хотели бы перейти к общему доказательству для сетей Хопфилда.
- Мы там не зря называли $\mathbf{e}$ вектором возбуждений — это действительно связано с системой из элементарных частиц.
- Сейчас мы вспомним вариационные методы и докажем, что сеть Хопфилда куда-нибудь сходится.

Вариационные методы

- В статфизике часто бывают распределения типа

$$p(\mathbf{x}) = \frac{1}{Z} e^{-\beta E(\mathbf{x}, J)}, \text{ где, например,}$$

$$E(\mathbf{x}, J) = -\frac{1}{2} \sum_{i,j} J_{ij} x_i x_j - \sum_i h_i x_i.$$

- Эта E — функция энергии системы элементарных частиц со спинами $\mathbf{x}$.

Приближение E

- Как нам обработать такую функцию?
- Будем её приближать более простым распределением:

$$Q(\mathbf{x}, \mathbf{a}) = \frac{1}{Z} e^{-\sum_i a_i x_i}.$$

- Качество приближения будем оценивать посредством *вариационной свободной энергии*

$$\beta \tilde{F} = \sum_{\mathbf{x}} Q(\mathbf{x}, \mathbf{a}) \ln \frac{Q(\mathbf{x}, \mathbf{a})}{e^{-\beta E(\mathbf{x}, J)}}.$$

- Это на самом деле средняя энергия E по распределению Q минус энтропия Q.
- Чем ближе приближение к p, тем меньше $\beta \tilde{F}$.

Приближение E через Q: энтропия

- В нашем конкретном случае энтропия Q — это сумма энтропий индивидуальных спинов

$$S_Q = \sum_{\mathbf{x}} Q \ln \frac{1}{Q} = \sum_i H_2(q_i) = \sum_i \left(q_i \ln \frac{1}{q} + (1-q) \ln \frac{1}{1-q} \right).$$

- Здесь q_i — вероятность того, что спин x_i равен $+1$, то есть

$$q_i = \frac{e^{a_i}}{e^{a_i} + e^{-a_i}} = \frac{1}{1 + e^{-2a_i}}.$$

Приближение E через Q: среднее по Q

- Среднее по Q тоже будет достаточно просто получить:

$$\sum_i Q(\mathbf{x}, \mathbf{a}) E(\mathbf{x}, J) = -\frac{1}{2} \sum_{i,j} J_{i,j} \bar{x}_i \bar{x}_j - \sum_i h_i \bar{x}_i,$$

где $\bar{x}_i = \frac{e^{a_i} - e^{-a_i}}{e^{a_i} + e^{-a_i}} = \tanh a_i = 2q_i - 1$.

Упражнение. Доказать эти формулы. Главное — то, что x_i и x_j в $J_{ij} x_i x_j$ независимы.

Минимизация

Теперь надо минимизировать вариационную свободную энергию

$$\beta \tilde{F} = \beta \left(-\frac{1}{2} \sum_{i,j} J_{i,j} \bar{x}_i \bar{x}_j - \sum_i h_i \bar{x}_i \right) - \sum_i H_2(q_i).$$

Упражнение. Взять частные производные и доказать, что минимум достигается в

$$a_k = \beta \left(\sum_i J_{ki} \bar{x}_i + h_k \right), \qquad \bar{x}_k = \tanh a_k.$$

От минимизации к алгоритму

- В этих уравнениях a_i выражаются через x_i и наоборот.
- Если пользоваться ими как итеративной процедурой, то $\beta\tilde{F}$ будет уменьшаться.
- Такая функция называется *функцией Ляпунова*. Если функция Ляпунова есть, то, значит, динамическая система точно сходится к точке или циклу, на котором функция Ляпунова константна.

Сети Хопфилда

- В сетях Хопфилда всё то же самое:

$$\beta \tilde{F}(\mathbf{x}) = -\beta \frac{1}{2} \mathbf{x}^\top \mathbf{W} \mathbf{x} - \sum_i H_2 \left(\frac{1 + x_i}{2} \right).$$

- Но это сильно зависит от условий задачи.

Упражнение.

1. Приведите пример сети Хопфилда с несимметричными весами, которая не сходится к одному состоянию.
2. Приведите пример сети Хопфилда с синхронными апдейтами, которая не сходится к одному состоянию.

Outline

Сети Хопфилда со временем

- Нехорошо, что мы зависим от того, синхронные апдейты или асинхронные.
- Поэтому можно на самом деле не зависеть, а считать реакцию нейронов функцией от времени.
- Будем считать, что $a_i(t) = \sum_j w_{ij} x_j(t)$ подсчитывается мгновенно, а нейрон реагирует по уравнению

$$\frac{d}{dt} x_i(t) = -\frac{1}{\tau}(x_i(t) - f(a_i)),$$

  где $f(a)$ — функция активации (tanh).
- Тогда, если матрица весов симметрична, эта динамическая система будет иметь ту же самую функцию Ляпунова.

Распознавание образов

- Сети Хопфилда применяют, например, для распознавания образов.
- При этом стабильные состояния системы — это образцы для распознавания, и работает так: при поступлении образа начинаем запускать сеть, пока не сойдётся.
- Если пытаться запихнуть слишком много образов, получаются проблемы: ложные стабильные состояния, неустойчивые стабильные состояния...

Задачи оптимизации

- А ещё можно попробовать приспособить сети Хопфилда для constraint satisfaction.
- Например, для задачи коммивояжёра на K городах можно рассмотреть сеть с K^2 нейронами, каждый из которых соответствует тому, что город i находится на j–ом месте пути.
- Веса должны обеспечивать, чтобы путь был правильный (отрицательные веса на нейроны в одной строке и столбце), а остальные соответствуют расстояниям.
- Но тут тоже надо аккуратно.

Thank you!

Спасибо за внимание!