Частичное обучение в логико-марковской сети в задаче извлечения временной информации из текста

© Фамхынг Д.К.

Волгоградский государственный технический университет
hungpdq@gmail.com

Аннотация

В работе описан метод использования логико-марковской сети для задачи извлечения временной информации в тексте на естественном языке. Предложенный метод является полным интегрированным решением, обеспечивающим строгие продукционные правила и правила с весом, указывающим уровень их доверия. Предложен новый алгоритм использования не аннотированных данных для повышения адекватности работы логико-марковской сети.

1 Введение

В связи с увеличением количества доступных электронных текстов требование к системам автоматической обработки и извлечения информации из документов на естественном языке для различных целей, таких, как добыча данных, значительно возросло. Например, нас, возможно, будет интересовать, когда и за сколько времени случилось некоторое событие. Эта задача связана с проблемой определения временных выражений в тексте, а также определения самого события. Например, дано следующее предложение:

«С 1-ого января в Москве подорожает проезд в общественном транспорте».

Исходя из этого предложения, мы можем определить, что «***подорожание»*** – это событие, и что оно случится с «***1-ого января»*.** После того как эта информация уже была извлечена, она может быть использована для создания более структурированной базы знаний, которую можно легко использовать в других системах обработки естественного языка (ОЕЯ), таких, как система поиска, система реферирования документов и вопросно-ответная система.

Задача извлечения временной информации из текста на естественном языке интенсивно исследуется в последние годы. Ее можно разделить на четыре подзадачи:

- Идентификация временных выражений;
- Идентификация событий в тексте;
- Связывание события с временным отчетом;
- Определение временных отношений между событиями в тексте.

Для решения этой задачи были проанализированы и используются различные подходы. Три основных подхода включают: традиционный подход, основанный на использовании правил, полученных от экспертов лингвистических знаний, подход, основанный на статистических методах, использующих какой-либо из алгоритмов машинного обучения, и гибридный подход. В работе [1] авторы поясняют преимущества и недостатки каждого из этих подходов. По сути, задача обработки естественного языка сложна из-за двусмысленности. Это вызывает трудности и ведет к неэффективности при применении статистических методов, так как большинство из них требует представить объект обучения в виде признаковых векторов. Аппарат логики, например, логика первого порядка или нечеткая логика, имеет широкие возможности для представления различных связей между явлениями в естественном языке, но он ограничен в способности обучения. Самые эффективные системы индуктивного обучения и вывода, такие как ALEPH [3], FOIL [4], Claudien, достигли не очень высокой адекватности.

В данной работе мы предлагаем новый метод для решения задачи извлечения временной информации с помощью аппарата логико-марковской сети (Markov logic networks), которая разработана в 2006 г. (ряд работ Домингоса и др. 2006-2008 гг.). Этот аппарат является вероятностным обобщением логики первого порядка, статистического обучения, позволяющего автоматически оценивать обоснованность выбранной модели явления и индуктивных правил, описывающих нестрогие зависимости между данными. Аппарат логико-марковской сети является самым удачным механизмом объединения мощности представления знания в традиционной двоичной логике и эффективности статистического обучения.

Статья состоит из четырех частей. Сначала мы описываем основные понятия марковской сети, логики первого порядка и логико-марковской сети. Далее будет подробно описано ее применение в

Труды 11[й] Всероссийской научной конференции «Электронные библиотеки: перспективные методы и технологии, электронные коллекции» - RCDL'2009, Петрозаводск, Россия, 2009.

решении нашей задачи. Наконец, мы представляем меру близости для временных отношений и их интеграцию в Марковской сети.

2 Логико-марковская сеть (MLNs)

Логико-марковская сеть (Markov logic networks) комбинирует логику первого порядка и марковскую сеть. В обычной базе знания на основе правил продукции (логике первого порядка) типа «If X is E then Y is A», если хотя бы одно утверждение не выполняется, то полнота БЗ нарушается. MLNs можно рассматривать как обобщение логики первого порядка, при которой, когда одна формула не выполняется, то БЗ имеет меньшую вероятность существования.

2.1 Марковская сеть (Markov network)

Марковская сеть - неориентированная вероятностная графовая модель, использующаяся для представления совместных распределений набора нескольких случайных переменных *X*. Формально марковская сеть состоит из следующих компонентов:

- Неориентированный граф G = (V, E), где каждая вершина v □ V является случайной переменной в X и каждое ребро {u,v} □ E представляет собой зависимость между случайными величинами u и v.
- Набор потенциальных функций (potential function) φ_k , одна для каждой клики в G. Функция φ_k ставит каждому возможному состоянию элементов клики в соответствие некоторое неотрицательное действительное число.

Совместное распределение набора *X* в марковской сети вычисляется по формулам:

$$P(X = x) = \frac{1}{Z}\prod_k \phi_k(x_{\{k\}}),$$

где $x_{\{k\}}$ представляет собой состояние случайных переменных в k-ой клике и Z является коэффициентом нормализации

$$Z = \sum_{x \in X}\prod_k \phi_k(x_{\{k\}}).$$

2.2 Логико–марковская сеть (MLNs)

Логико-марковская сеть представляет собой множество пар $\{(F_i,w_i)\}$, где:

F_i - формула логики первого порядка

w_i - действительное число

$\{(F_i,w_i)\}$ вместе с набором констант C = $(c_1,c_2,\ldots,c_{|C|})$ используются как шаблон для создания марковской сети $M_{L,C}$, содержащей:

- одну вершину для каждой возможной интерпретации (grounding) любого предиката в L. Данной вершине присваивается значение 1, если ее интерпретация верна, и 0 - в противном случае;
- один фактор для каждой интерпретации любой формулы F_i в L с соответствующим весом w_i.

Для пояснения, как работает MLNs, мы построим примерную модель для задачи фильтрации спама. Здесь мы имеем три предиката, *H(l)* означает «заголовок письма *l* длиннее, чем его содержание», *S(l)* означает «*l* является спам-письмом» и *Ad(l,k)* означает «письма *l* и *k* приходят с одного адреса». Правила описаны в таблице 1.

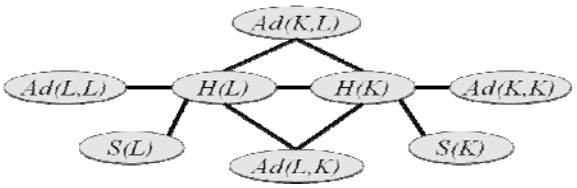

Рис 1. Марковская сеть, полученная из интерпретации формулы в таб. 1.

Распределение вероятностей возможного мира (БЗ) марковской сети $M_{L,C}$ определяется:

$$P(X = x) = \frac{1}{Z}\exp(\sum_i w_i n_i(x)) = \frac{1}{Z}\prod_i \phi_i(x_{\{i\}})^{n_i(x)}$$

где $n_i(x)$ является количеством верных интерпретаций (true grounding) формулы F_i в x , $x_{\{i\}}$ есть состояние (state) атомов, появляющихся в F_i и $\phi_i(x_{\{i\}}) = e^{w_i}$.

2.3 Вывод в MLNs

Допустим, что мы уже построили логико-марковскую сеть. Процесс вывода в ней часто требует и вероятностных, и детерминированных методов. Даны некоторые факты (evidences), требуется найти значения запрошенных предикатов для максимизации их частного распределения. Структура сети обычно бывает сложной, и в этом случае точный вывод является трудноразрешимой проблемой. Поэтому метод приблизительного рассуждения, такой как MCMC семплинг (Markov chain Monte Carlo sampling), станет хорошим выбором.

В связи с тем, что для решения нашей задачи мы используем и вероятностные правила, и строгие правила (deterministic dependencies), семплирование по Гиббсу ([2]), неэффективно обращается с детерминированными зависимостями, поэтому мы используем моделирование темперирования ([2]) для вывода.

Таблица 1- Спам-письмо

Утверждение	Логика 1-го порядка	Форма высказываний	Вес
Если заголовок письма длиннее чем его содержание, то это спам	$\forall l \quad H(l) \Rightarrow S(l)$	$\neg H(l) \vee S(l)$	1.1
Письма, приходящие с одного адреса либо являются спамом, либо нет	$\forall l \forall k \ \mathrm{Ad}(l,k) \Rightarrow S(l) \Leftrightarrow S(k)$	$\neg Ad(l,k) \vee S(h) \vee \neg S(l),$ $\neg Ad(l,k) \vee \neg S(h) \vee S(l)$	1.7 1.7

Таблица 2 – Предикаты для описания события

Предикаты	Объяснение	Константы
Tense(e,!t)	Временная форма события e	{Прошедшее, Настоящее, Будущее}
Imperfect(e,!p)	Видовая форма	{Совершенный, Несовершенный}
Aspect (e,!a)	Аспектуальный класс	{Моментальное событие, Развивающееся событие, Процесс и др.}
Modal(e,!m)	Модальность	{Возможность, Вероятность, Обязанность}
Polarity(e,!p)	Полярность	{Положительное, Отрицательное}
VerbClass(e,!v)	Форма имени события	{Глагол (*в личной форме и инфинитив*), Краткое прилагательное, Отпредикатное имя, Причастие, Деепричастие}

3 Модель извлечения временной информации в MLNs

В данной работе мы рассмотрим проблему определения временных отношений между событиями, описанными в источнике. Извлечение временных отношений заключается в определении взаимосвязей между событиями или между событиями и моментами времени [1]. В процессе вывода временных отношений возникает проблема, связанная с тем, что временная информация выражается в тексте на естественном языке явным или неявным образом. В любом случае остается неоспоримым тот факт, что события всегда связаны друг с другом или с моментом времени.

Как было отмечено в работе [1], на построение временного порядка влияет множество различных грамматических категорий, например, видо-временные формы глаголов, наречия времени, грамматические единицы (краткое прилагательное, отглагольное имя существительное). Кроме того, связь между событиями, отраженными в сложном предложении осуществляется с помощью союзов.

3.1 Предикаты

В нашей модели мы определяем различные предикаты для охвата этих явлений.

Для каждого события определены предикаты с разными грамматическими характеристиками: Tense(e,!t), Imperfect(e,!p), Aspect(e,!a), Class(e,!c), Modal(e,!m), Polarity(e,!p), VerbClass(e,!v),…

e – это событие,

знак «!» означает, что каждое событие e принадлежит только одному классу.

Мы также используем предикат HasVerb(e,+w) для указания, что событие e имеет слово w.

Для описания случая, когда специальные слова (союзы) присутствуют между событиями (мы считаем момент времени как специальное событие), мы приводим предикат HaveConnectedWord(e_1,e_2,+c), где e_1, e_2 – события, с есть союзное слово, например, *перед тем как, после, во время, с тех пор, когда, пока, как, в то время как,…*

Два события, имеющие одинаковый объект, выражаются предикатом HaveSameObject(e_1,e_2).

События, которые описываются в одном предложении, мы выражаем их предикатом InSameSentence(e_1,e_2).

Главным предикатом запроса является InGroup(e_1,e_2,g), означающий, что временное отношение между e_1,e_2 принадлежит группе *g*. Приняв классификацию временных отношений в работе [1], мы определяем девять групп временных отношений: {AFTER, BEFORE, DURING, INCLUDES, OVERLAPS, IS_OVERLAPPED, EQUALS, BEGIN, END}

В нашей работе мы рассмотрели два варианта описания предиката запроса: InGroup(e_1,e_2,!g) и InGroup(e_1,e_2,+g). Первый вариант означает, что временное отношение между e_1,e_2 принадлежит только одной из девяти возможных групп, в то время как второй разрешает мультигрупповой случай.

3.2 Формулы

Мы построили правила, чтобы описать нашу гипотезу о влиянии различных грамматических аспектов на временные отношения.

Для представления связи между видо-временной формой глаголов и временным отношением двух событий, мы предлагаем следующее правило:

Tense(e_1,+t_1) ^ Tense(e_2,+t_2) ^ Imperfect(e_1,+p_1) ^ Imperfect(e_2,+p_2) => InGroup(e_1,e_2,g)

Для описания влияния аспектуального класса, модальности и полярности на временное отношение двух событий, мы определяем следующее правило:

Aspect (e_1,+a_1) ^ Aspect (e_2,+a_2) ^ Modal(e_1,+m_1) ^ Modal(e_2,+m_2) ^ Polarity(e_1,+p_1) ^ Polarity (e_2,+p_2) => InGroup(e_1,e_2,g)

Когда последовательность событий обозначается с помощью союзных слов, тогда с большой вероятностью можно сказать, что эти слова играют важную роль в построении временного порядка. Для этого случая мы используем простое правило:

HaveConnectedWord(e1,e2,+c) => InGroup(e_1,e_2,g)

Причинно-следственное отношение часто приводит к тому, что одно событие является следствием другого события, и поэтому произошло после него. Например:

«Свет выключился. Комната была темна».

Для этого мы введем новый предикат Casual(e_1,e_2), обозначающий, что событие e_1 является причиной e_2. Мы также добавим новое правило:

Casual(e_1,e_2) => InGroup(e_1,e_2, «BEFORE»)

Иногда сами имена событий, т.е. слова, обозначающие события, влияют на их временной порядок. Рассмотрим пример:

«Я упал на пол. Кто-то меня толкнул»

В этом примере между событиями нет никакого союзного слова. Но, исходя из знания о познаваемом мире, мы знаем, что падение всегда происходит после толчка, и сможем сделать правильное заключение об их временном отношении. Для описания этого появления мы определяем правило:

HasVerb(e_1,+w_1) ^ HasVerb(e_2,+w_2) => InGroup(e_1,e_2,g)

Детерминированные правила

MLNs допускает строгие правила, и этим правилам будет присваиваться степень доверия (веса) с бесконечным значением. Это можно рассмотреть как продукционные правила, которые всегда выполняются. Мы сформулировали 5 групп экспертных правил для разных случаев проявления событий.

Транзитивные правила

В других работах, опубликованных ранее, транзитивные правила используются для повышения количества найденных временных отношений. Но во всех этих работах это происходит как пост-процесс только после процесса классификации [6]. Это приводит к несогласованности в базе знания. В работе [7] авторы предложили метод, гарантирующий глобальную согласованность в базе знания с помощью линейного целочисленного программирования, но ограничились только двумя возможными отношениями. С помощью MLNs мы интегрируем эти правила в процессе вывода, следовательно, гарантируются согласованность и эффективность при выводе.

Некоторые транзитные правила являются строгими, некоторые выполняются с большой вероятностью, поэтому имеет вес с большим значением, а другие являются «мягкими» правилами, например:

InGroup(e_1,e_2, BEFORE) ^ InGroup(e_2,e_3, BEFORE) => InGroup(e_1,e_3, BEFORE)

InGroup(e_1,e_2, BEFORE) ^ InGroup(e_2,e_3, OVERLAPS) => InGroup(e_1,e_3, BEFORE)

InGroup(e_1,e_2, BEFORE) ^ InGroup(e_2,e_3, DURING) => InGroup(e_1,e_3, OVERLAPS)

4 Интеграция частичного обучения в MLNs

В задаче обработки естественного языка часто имеет место такая ситуация, что неаннотированных данных достаточно много и они легко собираются. Аннотирование этих данных очень дорого и требует колоссальных усилий. В данной работе мы предложили алгоритм, который использует неаннотированные данные для повышения эффективности работы MLNs.

Алгоритм основан на простой идее, что два близких объекта будут находиться в одной группе.

Кластеризация

На основе меры сходства мы используем метод кластеризации (k-NN и KNN-SVM [5]) для обучения корпуса, комбинирующего аннотированные и неаннотированные данные. За счет неаннотированного корпуса плотность данных повышается, и аккуратность кластеризации увеличивается. Обучаемые выборки, следовательно, разделятся на кластеры.

Применим этот алгоритм в решении нашей задачи. После кластеризации мы добавляем в базу знаний нашей MLNs новый предикат Label(e_1,e_2,e_3,e_4), указывающий на то, что временное отношение между парами событий (e_1,e_2) и (e_3,e_4) принадлежит одному кластеру.

Интеграция частичного обучения в MLNs

Обучение и вывод в MLNs реализуются посредством максимизации псевдо-функции правдоподобия (pseudo-likelihood function), являющейся маргинальной вероятностью запрошенных предикатов при некоторых заданных свидетельствах. Интеграция частичного обучения в MLNs поэтому затруднена. Для этого мы добавляем правила:

Label(e_1,e_2,e_3,e_4) ^ InGroup(e_1,e_2,+g) => InGroup(e_3,e_4,+g).

5 Заключение

В работе предложен новый подход к решению задачи извлечения временной информации с помощью аппарата логико-марковской сети. Интеграция априорных знаний всегда является большой проблемой в обработке естественного языка. Логико-марковская сеть дает естественный способ решения с поддержкой строгих правил и «мягких» правил, за счет которых смягчает условие согласованности базы знаний. Однако в логико-марковской сети никак нельзя влиять на процесс выбора обучаемой выборки, потому что вес для всех означенных формул (grounding formula) одной формулы зафиксирован. Предложенный в данной

работе метод использования меры близости дает возможность регулировать целевую функцию и поэтому влиять на определение значимости обучаемой выборки.

Литература

[1] Заболеева-Зотова А.В., Фамхынг Д.К., Захаров С.С. Гибридный подход к обработке временной информации в тексте на русском языке // Труды одиннадцатой национальной конференции по искусственному интеллекту с международным участием - **КИИ** 2008.

[2] Richardson, M., Domingos, P. Markov Logic Networks. Machine Learning, volume 62 , pages 107–136, 2006

[3] Srinivasan, A. (2001). The Aleph manual. http://web.comlab.ox.ac.uk/oucl/research/areas/machlearn/Aleph/.

[4] Landwehr, N., Kersting, K., & Raedt, L. D. Integrating Naive Bayes and FOIL. Journal of Machine Learning Research, volume 8, pages 481–507, 2007.

[5] Hao Zhang. SVM-KNN: Discriminative Nearest Neighbor Classification for Visual Category Recognition. In proceeding of IEEE Computer Society Conference on Computer Vision and Pattern Recognition, 2006, Vol. 2, P. 2126-2136.

[6] Temporal Awareness and Reasoning Systems http://www.timeml.org/site/tarsqi/index.html

[7] Bramsen, P. and Deshpande, P. Inducing Temporal Graphs. In Proceedings of EMNLP-06, 2007.

Semi-supervised learning with Markov Logic Networks and application to temporal information processing

Hung Pham D.Q.

This paper addressed the problem of temporal information extraction from text. We proposed a solution using Markov Logic Networks. The knowledge is presented in first-order logic production rules. We used both hard clauses and clauses that associated with weights to indicate their confidence. A new co-training algorithm was proposed that take benefit of unlabeled data to boost accuracy of Markov logic networks.