دانشکده مهندسی کامپیوتر

بررسی روش‌های توأم برچسب‌زنی و تجزیه‌ی وابستگی

سمینار کارشناسی ارشد

در رشته‌ی مهندسی کامپیوتر-گرایش هوش مصنوعی و رباتیک

نام دانشجو

عاطفه پاکزاد

استاد راهنما

دکتر بهروز مینایی بیدگلی

آبان ۱۳۹۲

دانشکده مهندسی کامپیوتر

بررسی روش‌های توأم برچسب‌زنی و تجزیه‌ی وابستگی

سمینار کارشناسی ارشد

در رشته‌ی مهندسی کامپیوتر-گرایش هوش مصنوعی و رباتیک

نام دانشجو

عاطفه پاکزاد

استاد راهنما

دکتر بهروز مینایی بیدگلی

آبان ۱۳۹۲

لازم می‌دانم از زحمات جناب آقای دکتر مینایی نهایت تشکر و قدردانی را به عمل آورم. و از راهنمایی‌های مهندس خلاش نیز سپاسگزارم.

چکیده

برچسب اجزای سخن[1] از ویژگی‌های ضروری در تجزیه‌ی وابستگی است. در اکثر پژوهش‌های انجام شده، تجزیه‌ی وابستگی با فرض وجود برچسب‌های دستی[2] و یا حالتی که با برچسب‌های پیش‌بینی شده[3] توسط یک برچسب‌زن جایگزین شده مورد بررسی قرار گرفته است. در این رویه دو وظیفه برچسب‌زنی اجزای سخن و تجزیه‌ی وابستگی به صورت مستقل مورد مطالعه قرار می‌گیرد که این نحوه‌ی برخورد با مسئله، مشکل انتشار خطا را بوجود می‌آورد به عنوان مثال اگر اسمی به اشتباه برچسب صفت خورده باشد دیگر در تجزیه‌ی وابستگی نمی‌تواند نقش مفعول را به خود بگیرد.

در حال حاضر اکثر تجزیه‌گرهای فعلی فرض می‌کنند که کلمات ورودی قبل از شروع تجزیه از لحاظ ساخت‌واژی[4] با استفاده از برچسب‌زن اجزای سخن ابهام‌زدایی شده‌اند [۱]. روش پایپلاین سنتی برای برچسب‌زنی اجزای سخن و تجزیه‌ی وابستگی از مسئله‌ی انتشار خطا رنج می‌برد[۲]. برای حل مشکل انتشار خطا، مدل‌های توأمی برای بهینه‌سازی برچسب‌زنی اجزای سخن و تجزیه‌ی وابستگی ارائه شده است. مدل‌های توأم رویکردی رایج و تاثیرگذار برای انجام همزمان وظایف مشابه می‌باشند.

اطلاعات نحوی بعضی از ابهامات اجزای سخن که برای مدل‌های ترتیبی برچسب‌زنی اجزای سخن وجود دارد را برطرف می‌کند و از طرف دیگر برچسب‌های دقیق اجزای سخن باعث بهبود تجزیه‌ی وابستگی می‌شود. ابهام‌زدایی نحوی و ساخت‌واژی به‌ویژه برای زبان‌هایی که از لحاظ ساخت‌واژی غنی[5] هستند بسیار مهم است، و همچنین تعامل قابل ملاحظه‌ای بین نحو و ساخت‌واژی وجود دارد که نمی‌توان یکی را بدون در نظر گرفتن دیگری ابهام‌زدایی کرد[۱].

مشاهدات اخیر نشان می‌دهد که دقت تجزیه را می‌توان تا حد زیادی با بهینه‌سازی توأم برچسب‌زنی اجزای سخن و تجزیه‌ی وابستگی بهبود داد. البته، وظیفه‌ی برچسب‌زنی اجزای سخن سود زیادی از چارچوب توأم نمی‌برد. دلیل اصلی این است که ویژگی‌های نحوی بر ویژگی‌های اجزای سخن در طول بهینه‌سازی تسلط داشته‌اند و مدل‌های توأم تنها آن دسته از برچسب‌های اجزای سخن را پیشنهاد می‌دهند که از نقطه نظر تجزیه مطلوب هستند[۳].

کلمات کلیدی: برچسب‌زنی اجزای سخن، تجزیه‌ی وابستگی، مدل‌های توأم، تجزیه‌ی مبتنی بر گذار، تجزیه‌ی مبتنی بر گراف، مدل‌های پشته‌ای

---

[1] POS tags
[2] Gold POS tag
[3] Predicted POS tag
[4] Morphology
[5] Morphologically-Rich Languages (MRLs)

فهرست مطالب

فهرست شکل ها

فهرست جدول ها

فصل اول

مقدمه

۱-مقدمه

۱-۱-شرح مسئله

زبان‌شناسی[6] علمی است که به بررسی نظام‌مند زبان می‌پردازد. از دیدگاه چامسکی اصلی‌ترین مسئله در زبان‌شناسی فهم زبان است. یکی از پیچیدگی‌های فهم زبان، وابستگی اجزای زبان به بافت[7] است. با فراگیر شدن استفاده از رایانه، تمایل به استفاده از رایانه برای فهم زبان‌های طبیعی انسانی به وجود آمد که این امر منجر به ایجاد زمینه‌ی تازه‌ای تحت عنوان زبان‌شناسی رایانه‌ای[8] یا پردازش زبان طبیعی[9] شد. در این حوزه فهم زبان معادل تجزیه‌ی[10] زبان است. در تجزیه‌ی نحوی ارتباط بین واژه‌های جمله یافت می‌شود. دستور وابستگی یکی از نظریه‌های زبان‌شناختی است که در بررسی نحو و دستور زبان به کار می‌رود. تجزیه‌ی وابستگی نیز یک الگوی مبتنی بر دستور وابستگی برای تجزیه و تحلیل خودکار جملات است. تجزیه‌ی وابستگی در حوزه‌های مختلفی مانند استخراج روابط، ترجمه‌ی ماشینی، تولید واژه‌های مترادف و تقویت منابع واژگانی کاربرد دارد[٤].

برچسب‌زنی اجزای سخن عمل انتساب برچسب‌های واژگانی به کلمات و نشانه‌های تشکیل دهنده‌ی یک متن است به صورتی که این برچسب‌ها نشان دهنده‌ی نقش کلمات و نشانه‌ها در جمله باشند. درصد بالایی از کلمات از نقطه نظر برچسب اجزای سخن دارای ابهام هستند زیرا کلمات در جایگاه‌های مختلف برچسب‌های اجزای سخن متفاوتی دارند. بنابراین برچسب‌زنی اجزای سخن عمل ابهام‌زدایی از برچسب‌ها با توجه به بافت مورد نظر است. برچسب‌زنی اجزای سخن عملی اساسی برای بسیاری از حوزه‌های دیگر پردازش زبان طبیعی از قبیل ترجمه ماشینی، خطایاب و تبدیل متن به گفتار و تجزیه‌ی وابستگی می‌باشد[٥].

دو مقوله‌ی برچسب‌زنی اجزای سخن و تجزیه‌ی وابستگی هر کدام به صورت جداگانه در مقالات مختلف مورد بررسی قرار گرفته‌اند. از آنجایی که برچسب‌های اجزای سخن به عنوان یک پیش‌نیاز نقش تعیین‌کننده‌ای در تجزیه‌ی وابستگی دارند و نیز کمکی که اطلاعات نحوی به برچسب‌زنی اجزای سخن می‌کند روش‌های توأم برچسب‌زنی اجزای سخن و تجزیه‌ی وابستگی برای بهبود دقت برچسب‌زنی و نیز دقت تجزیه‌ی وابستگی پیشنهاد شده‌اند.

---

[6] Linguistic
[7] Context
[8] Computational linquistics
[9] Natural Language Processing(NLP)
[10] Parsing

۱-۲-ساختار فصل‌های آینده

در فصل دو به طور خلاصه به برچسب‌زنی اجزای سخن و تجزیه‌ی وابستگی و روش‌های حل آن پرداخته شده است و در مورد روش پایپلاین برچسب‌زنی اجزای سخن و تجزیه‌ی وابستگی توضیح داده شده، سپس ضرورت استفاده از مدل‌های توأم بیان گردیده است. در فصل سوم تعدادی از مدل‌های توأم ارائه شده در سه دسته‌ی مدل‌های توأم مبتنی بر گذار، مدل‌های توأم مبتنی بر گراف و مدل‌های توأم ترکیبی مورد بررسی قرار گرفته‌اند. در فصل چهارم ضمن جمع‌بندی مباحث مطرح شده در فصول گذشته، نقاط ضعف این روش‌ها مطرح شده، تا کارهای آینده بر این اساس جهت‌دهی شود.

فصل دوم
تعاریف و مفاهیم مبنایی

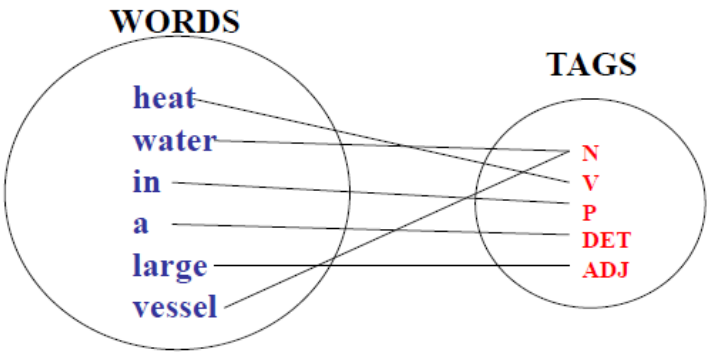
۲-تعاریف و مفاهیم مبنایی

مقدمه

در این فصل دلیل پرداختن به موضوع سمینار و صورت مساله آن مورد بررسی قرار می‌گیرد. برای این منظور، ابتدا مقدماتی در رابطه با موضوعات برچسب‌زنی اجزای سخن و تجزیه‌ی وابستگی ارائه می‌شود سپس مدل پایپلاین برچسب‌زنی اجزای سخن و تجزیه‌ی وابستگی توضیح داده می‌شود و در انتهای فصل ضرورت استفاده از مدل‌های توأم برای برچسب‌زنی و تجزیه‌ی وابستگی بیان می‌گردد.

۲-۱- برچسب‌زنی اجزای سخن

برچسب‌زنی اجزای سخن، عمل انتساب برچسب‌های واژگانی به کلمات و نشانه‌های تشکیل دهنده‌ی یک متن است به صورتی که این برچسب‌ها نشان‌دهنده‌ی نقش کلمات و نشانه‌ها در جمله باشند. برخی از سیستم‌های برچسب‌زنی هنگام انتساب برچسب به کلمات، یک یا چند برچسب به کلمه منتسب می‌کنند. اگر به کلمه چند برچسب منتسب شود، برچسب کلمه در اصطلاح مبهم است و نیاز به ابهام‌زدایی دارد. برخی سیستم‌های برچسب‌گذاری سعی می‌کنند به هر کلمه تنها یک برچسب منتسب کنند. خروجی این سیستم‌ها مبهم نمی‌باشد و نیاز به ابهام‌زدایی وجود ندارد.



شکل ۱-فرآیند نسبت دادن اجزای سخن به هر کلمه در یک جمله

در زیر نمونه‌ای از یک مجموعه برچسب برای زبان فارسی ارائه شده است[۵]:

ADV	Adverb	قید
AJ	Adjective	صفت
CL	Classifier	شاخص
CONJ	Conjunction	حرف ربط
DET	Determiner	حرف تعریف
INT	Interjection	حرف صوت
N	Noun	اسم
NUM	Number	عدد
P	Preposition	حرف اضافه
POSTP	Postposition (را)	حرف اضافه پسین
PRO	Pronoun	ضمیر
PUNC	Punctuation	جدا کننده
RES	Residual:Arabic and Latin words, etc	متفرقه
V	Verb	فعل

شکل ۲- یک مجموعه‌ی برچسب برای زبان فارسی

۲-۱-۱-انواع روش‌های برچسب‌زنی اجزای سخن

برچسب‌گذاری قانون‌محور[11]: در این روش برچسب‌گذاری دو مرحله‌ی زیر را خواهیم داشت:

- به هر واژه با توجه به واژه‌نامه‌ی موجود، فهرستی از برچسب‌های ممکن تعلق می‌گیرد؛
- با استفاده از فهرست بزرگی از قوانین ابهام‌زدایی که به صورت دستی جمع‌آوری شده است، تعداد برچسب‌ها برای هر واژه به یک برچسب تقلیل می‌یابد.

برچسب‌گذاری تصادفی[12]: در این روش، از روش‌های احتمالی و آماری استفاده می‌شود. معروف‌ترین روش برای برچسب‌گذاری تصادفی استفاده از الگوی پنهان مارکوف[13] است. به برچسب‌گذاری که با الگوی پنهان مارکوف کار می‌کند، برچسب‌گذاری مارکوفی گفته می‌شود، روش‌های دیگری برای برچسب‌گذاری تصادفی مانند روش بیشینه‌ی آنتروپی[14] ، میدان‌های تصادفی شرطی[15] و پرسپترون نیز وجود دارد.

[11] Rule-based Tagging
[12] Stochastic Tagging
[13] Hidden Markov Model(HMM)
[14] Maximum Entropy
[15] conditional random fields

برچسب‌گذاری انتقال محور[16]: در این برچسب‌گذاری، از روش یادگیری انتقال محور[17] استفاده می‌شود. در یادگیری انتقال محور سه مرحله‌ی زیر انجام می‌شود:

- به هر واژه برچسبی که دارای بیش‌ترین احتمال است، تعلق می‌گیرد؛
- هر انتقال ممکن در بین نشانه‌ها مورد آزمون قرار می‌گیرد و انتقالی که باعث بهبود در برچسب‌گذاری می‌شود، انتخاب می‌شود؛
- با استفاده از انتقال‌های صورت گرفته و قوانین موجود برچسب‌گذاری واژگان بازبرچسب‌گذاری[18] می‌شوند [٦].

برچسب‌زنی اجزای سخن یک مسئله‌ی برچسب‌زنی معمول است. بسیاری از مدل‌ها نظیر بیشینه‌ی آنتروپی [٧] و میدان‌های تصادفی شرطی[19] (CRF)[8] و پرسپترون [٩] به طور موفقیت‌آمیزی به مسائل برچسب‌زنی دنباله[20] اعمال شده است [١]. در اینجا مدل پرسپترون به اختصار مورد بررسی قرار می‌گیرد.

در ابتدا، به عنوان یک مدل خطی، پرسپترون ساده، سریع و موثر است و در دقت برچسب‌زنی با CRF قابل مقایسه است اما به زمان آموزشی بسیار کمتری احتیاج دارد. پرسپترون به صورت موفق به تجزیه‌ی وابستگی اعمال شده است.

در پرسپترون، امتیاز دنباله‌ی برچسب به صورت زیر است:

$$Score_{pos}(x,t) = w_{pos}.\, f_{pos}(x,t)$$

$f_{pos}(x,t)$ به بردار ویژگی اشاره می‌کند و w_{pos} بردار وزن متناظر با آن است.

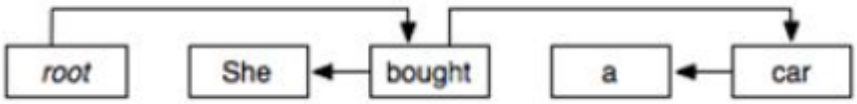
برای ویژگی‌های برچسب‌زنی اجزای سخن سه مجموعه‌ی ویژگی در نظر گرفته می‌شوند: ویژگی‌های اجزای سخن یونیگرام، بایگرام و تریگرام. برای اختصار به این سه مجموعه به صورت $w_i t_i$ و $t_{i-1}t_i$ و $t_{i-2}t_{i-1}t_i$ اشاره می‌شود. با توجه به w_{pos} الگوریتم ویتربی[21] برای به‌دست آوردن دنباله‌ی برچسب‌زنی بهینه در نظر گرفته شده است[١].

[16] Transformation-based Tagging
[17] Transformation-based Learnin(TBL)
[18] Re- Tag
[19] Conditional Random Fields
[20] sequence labeling
[21] Viterby

۲-۲- تجزیه‌ی وابستگی

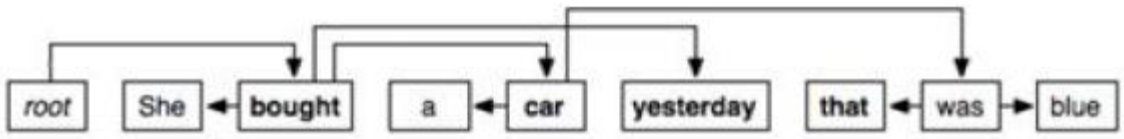
در مرجع [٤] که روش‌های تجزیه را مورد بررسی قرار داده است آمده است: تجزیه‌ی وابستگی راهی برای تجزیه‌ی نحوی زبان طبیعی است که به صورت خودکار به تجزیه و تحلیل ساختار وابستگی جملات پرداخته و برای هر جمله‌ی ورودی یک گراف وابستگی ایجاد می‌کند. در ساختار وابستگی هر گره نشان‌دهنده‌ی واژه‌های جمله است و وابستگی‌ها روابط نحوی و مفهومی گره‌ها را بازنمایی می‌کنند. در ساختار وابستگی، هر واژه حداکثر وابسته به یک واژه‌ی دیگر می‌تواند باشد. این بدان معنی است که این ساختار را می‌توان به صورت گراف جهت‌دار بازنمایی کرد که در آن گره‌ها معادل واژه‌ها و یال‌ها معادل روابط وابستگی هستند. یال‌ها می‌توانند با انواع وابستگی خاصی برچسب‌دار شوند یا بدون برچسب ایجاد شوند. گراف‌های وابستگی را از نظر نوع روابط وابستگی به دو دسته تقسیم می‌کنند:

۱- گراف‌های وابستگی افکنشی[۲۲]: اگر تمام واژه‌های جمله در یک خط چیده شود و گراف وابستگی بدون یال با تقاطع (یال با هم‌پوشانی) باشد، گراف وابستگی را افکنشی یا انعکاسی می‌نامند.



شکل ۳- مثالی از درخت وابستگی افکنشی

۲- گراف وابستگی غیرافکنشی[۲۳]: در این گراف‌ها حداقل دو یال که با یکدیگر همپوشانی داشته باشند، وجود دارد. در زبان‌های با ترتیب آزاد واژه‌ها، ویژگی غیرافکنشی پدیده‌ی رایجی است، زیرا محدودیت‌های بالقوه نسبی روی وابستگی‌ها خیلی کمتر انعطاف‌پذیر هستند.



شکل ٤- مثالی از درخت وابستگی غیرافکنشی

22 Projective dependency graph
23 Non-projective dependency graph

۲-۲-۱- روش‌های حل مسئله‌ی تجزیه‌ی وابستگی

روش‌های حل مسئله تجزیه‌ی وابستگی به دو دسته مبتنی بر دستور و مبتنی بر داده تقسیم می‌شوند که روش‌های مبتنی بر داده به دلیل ماهیت مستقل از زبان و دقت بالاتر بیش‌تر مورد استفاده قرار می‌گیرند. الگوریتم‌های موجود در این دسته به دو گروه «تجزیه مبتنی بر گذار» و «تجزیه مبتنی بر گراف» تقسیم می‌شوند[٤].

در الگوی تجزیه روش‌های مبتنی بر گذار ، گراف‌های وابستگی با اعمال رشته‌ای از اقدام‌ها یا گذارها تولید می‌شود. هر دو الگوریتم یادگیری و استنتاج بر پایه‌ی پیش‌بینی گذار درست مبتنی بر حالت جاری و یا تاریخچه است [٤]. به مسئله‌ی تجزیه‌ی وابستگی به دید جستجوی قطعی توسط یک سامانه‌ی گذار یا ماشین حالت نگاه می‌شود که توسط یک الگوی احتمالی هدایت می‌شود تا گذار بعدی را پیش‌بینی کند. در این مدل‌ها مسئله را با اقدام‌های ابتدایی تجزیه، پارامتری می‌کنند و معمولا از الگوریتم افزایشی جستجوی پرتو[٢٤] بهره می‌برند[٢, ١٠].

در الگوی تجزیه روش‌های مبتنی بر گراف ، تابع امتیاز یا احتمال روی مجموعه‌ی یال‌های ممکن و کل گراف وابستگی تعریف می‌شود. بر همین اساس در زمان یادگیری پارامترهای این تابع تخمین زده شده و در زمان تجزیه به دنبال گرافی می‌گردد که مقدار این تابع را بیشینه کند. تفاوت اصلی این تجزیه‌کننده‌ها در موارد زیر است[٤] :

١) نوع و ساختار تابع امتیازدهی (الگو)

٢) الگوریتم جستجو که بهترین تجزیه را پیدا می‌کند (استنتاج)

٣) روش تخمین پارامترهای تابع (یادگیری)

مدل‌های مبتنی بر گراف مسئله‌ی تجزیه را با استفاده از ساختار گراف وابستگی و به طور کلی استفاده از برنامه‌نویسی پویا برای استنتاج پارامتری می‌کنند. علی رغم تفاوت‌های چشمگیر در ساختار مدل، تجزیه‌گرهای مبتنی بر گراف و مبتنی بر گذار دقت به‌روز[٢٥] را می‌دهند.

یکی دیگر از روش‌ها، روش‌های ترکیبی یا ترکیب دو تجزیه‌کننده است. [٤] این مدل‌ها را به صورت زیر توضیح داده است: الگوهای ترکیبی به دو دسته تقسیم شده است: ١-الگوهایی که تجزیه‌کننده‌های

---

[٢٤] Incremental beam search

[25] state-of-the-art accuracy

پایه‌ای[۲۶] را در زمان یادگیری ترکیب می‌کنند.۲-الگوهایی که تجزیه‌کننده‌های پایه‌ای به صورت مستقل آموزش می‌بینند و سپس در زمان تجزیه ترکیب می‌کنند.

در تجزیه‌ی مبتنی بر گذار، الگوی تجزیه بر اساس گذارها از یک حالت به حالت دیگر در یک ماشین حالت انتزاعی تعریف می‌شود. آموزش این الگو نوعا با استفاده از تکنیک‌های رده‌بندی استاندارد انجام می‌شود که در آن‌ها پیش‌بینی یک گذار از میان مجموعه‌ی گذارهای ممکن در تاریخچه‌ی حالت، آموخته می‌شود. استنتاج به صورت محلی است زیرا چنین سیستم‌هایی با حالت‌های اولیه‌ی محدود شروع به کار کرده و به صورت حریصانه گراف را با انتخاب گذارها با بیش‌ترین امتیاز در هر حالت می‌سازند و تا زمانی که به شرط خاتمه برسد، ادامه می‌دهند.

در تجزیه‌ی مبتنی بر گراف، الگوی تجزیه بر اساس زیر گراف‌های وابستگی تعریف می‌شود. آموزش به صورت سراسری است که الگو به صورت امتیاز عمومی گراف‌های درست، در مقابل نادرست‌ها آموزش می‌بینند. استنتاج نیز سراسری است، زیرا تلاش می‌کند گراف با بیش‌ترین امتیاز را از بین مجموعه تمام گراف‌ها بیابد.

در هر کدام از این روش‌ها خطاهای متفاوت و مکملی وجود دارد که منجر به دسته‌بندی جداگانه آن‌ها شده است. بر همین اساس الگوی ترکیبی پشته‌سازی ارائه شده است که ترکیب را در زمان یادگیری انجام می‌دهد و دو سیستم مکمل می‌توانند از یکدیگر بیاموزند. اساس این ترکیب مبتنی بر خصوصیات است. هر دو الگو از یک تابع امتیاز $\mathrm{s}: \mathrm{X} \rightarrow R$ استفاده می‌کنند با این تفاوت که X در دامنه‌های متفاوتی تعریف می‌شود. برای الگوی مبتنی بر گراف X مجموعه‌ی یال‌های وابستگی $(\mathrm{i,j,l})$ است و برای الگوی مبتنی بر گذار X مجموعه‌ی جفت حالت-گذار ممکن $(\mathrm{c,t})$ است. در هر دو حالت ورودی با بردار خصوصیت k بعدی $f: X \rightarrow R^k$ بازنمایی می‌شود. در یک تجزیه‌کننده‌ی پشته‌سازی بردار خصوصیات یک الگو، که به آن الگوی پایه‌ای می‌گویند، توسط تعداد خاصی از خصوصیات تولید شده توسط الگوی دیگر که به آن الگوی راهنما[۲۷] شناخته می‌شود، توسعه داده می‌شود. خصوصیات اضافه شده را نیز خصوصیات راهنما[۲۸] می‌نامند. نسخه‌ای از الگوی پایه‌ای که توسط بردار خصوصیات توسعه یافته آموزش دیده است را الگوی هدایت شده[۲۹] گویند. ایده‌ی اصلی این است که الگوی هدایت شده باید قادر باشد

[26] Base parser
[27] Guide model
[28] Guide features
[29] Guided model

که بیاموزد چه مواقعی باید به خصوصیات راهنما اطمینان کرد تا از نقاط قوت الگوی راهنما بهره برد. در این صورت کارایی تجزیه کننده می‌تواند نسبت به الگوی پایه‌ای بهتر شود.

یادگیری پشته‌ای یک رویکرد معمول برای یکپارچه‌سازی مدل است. ایده‌ی یادگیری پشته‌ای این است که شامل دو سطح پیش‌گویی کننده است: سطح ۰ شامل یک تا تعداد بیشتری پیش‌بینی کننده $g_1, \dots g_k (K \geq 1): R^d \to R$ است و سطح یک شامل یک پیش‌بینی کننده‌ی $h: R^{d+k} \to R$ است. هر پیش‌بینی کننده‌ی g_k سطح صفر ورودی $x \epsilon R^d$ را دریافت می‌کند و پیش‌بینی $g_k(x)$ را به عنوان خروجی می‌دهد. پیش‌بینی کننده‌ی سطح یک ورودی $\langle x, g_1(x), \dots, g_k(x) \rangle$ است و یک پیش‌بینی نهایی $h(x, g_1(x), \dots, g_k(x))$ را به عنوان خروجی می‌دهد[۱۱].

۲-۲-۱-۱- تجزیه‌ی وابستگی مبتنی بر گذار

در [۱۰] سیستم گذار را به صورت زیر توضیح داده است: مجموعه‌ی P برچسب‌های اجزای سخن و مجموعه‌ی D برچسب‌های وابستگی داده شده است، یک درخت وابستگی برچسب‌دار برای یک جمله‌ی $x = w_1, \dots, w_n$ یک درخت جهت‌دار $T=(V_x, A)$ با توابع برچسب‌زنی π و δ است که :

۱- $V_x = \{0,1, \dots, n\}$ مجموعه‌ای از گره‌هاست.

۲- $A \subseteq V_x \times V_x$ مجموعه‌ای از کمان‌هاست.

۳- $\pi : V_x \to P$ تابع برچسب‌زنی برای گره‌هاست.

٤- $\delta : A \to D$ تابع برچسب‌زنی برای کمان‌هاست.

٥- 0 ریشه‌ی درخت است.

مجموعه‌ی V_x گره‌ها، مجموعه‌ی اعداد صحیح مثبت رو به بالا و شامل n است، که هر کدام نظیر موقعیت خطی یک کلمه در جمله به علاوه یک ریشه‌ی مصنوعی اضافی 0 هستند. مجموعه‌ی A کمان‌های مجموعه‌ی جفت‌های (i,j) است که i گره‌ی سر و j گره‌ی وابستگی است. توابع π و δ یک برچسب اجزای سخن یکتا را به هر گره/کلمه و یک برچسب وابستگی یکتا را به هر کمان نسبت می‌دهد.

یک سیستم گذار برای تجزیه‌ی وابستگی به عنوان چهارتایی $S=(C,T,c_s,C_t)$ تعریف می‌شود که:

۱- C مجموعه‌ی ترتیب‌هاست.

۲- T مجموعه‌ای از گذارهاست که هر کدام از آن‌ها یک (جزء) تابع است. $t : C \to C$

۳- c_s یک تابع مقداردهی اولیه است، نگاشت یک جمله‌ی x به یک ترتیب $c \in C$.

٤- $C_t \subseteq C$ مجموعه‌ی ترکیب‌های نهایی است.

یک دنباله‌ی گذار برای هر جمله‌ی x در S یک دنباله‌ی جفت‌های ترتیب-گذار

$C_{0,m} = [(c_0, t_0), (c_1, t_1), \dots, (c_m, t_m)]$ است که $c_0 = c_s(x), t_m(c_m) \in C_t$ و

$t_i(c_i) = c_{i+1} (0 \leq i <$m) است.

مجموعه‌ی C ترتیب‌ها که مجموعه‌ی همه‌ی پنج‌تایی‌های $c = (\sum, B, A, \pi, \delta)$ است در نظر گرفته می‌شود. (پشته)$\sum$ و B(بافر) لیست‌های فرعی منفصل گره‌های V_x بعضی جمله‌های x است، و A مجموعه‌ی کمان‌های وابستگی روی V_x است و π و δ توابع برچسب‌زنی‌ای هستند که در بالا تعریف شد. ترتیب ابتدایی برای جمله‌ی x در نظر گرفته می‌شود که $x = w_1, \dots, w_n$ به صورت $c_s(x) =$ $([0], [1, \dots, n], \{\ \}, \perp$ و $\perp)$ که $\perp$ تابعی است که برای همه‌ی آرگومان‌ها تعریف نشده است، و در مجموعه‌ی C_t ترتیب‌های نهایی درنظر گرفته می‌شود تا مجموعه‌ی همه‌ی ترتیب‌های فرم $c=([0],[],A,\pi,\delta)$ (برای هر A, π, δ) باشد. درخت وابستگی برچسب خورده‌ی تعریف شده برای x با $c = (\sum, B, A, \pi, \delta)$ درخت (V_x, A) با توابع برچسب گذاری π, δ است که به صورت TREE (x,c) نوشته می‌شود.

مجموعه‌یT گذارها در شکل ۵ نشان داده شده است. گذارهای $\text{LEFT} - \text{ARC}_d$ و $\text{RIGHT} - \text{ARC}_d$ هر دو یک کمان (با برچسب وابستگی d) بین دو گره‌ی روی پشته ایجاد می‌کنند و این گره‌ها با گره‌ی سر کمان جدید (که برای $\text{LEFT} - \text{ARC}_d$ راست‌ترین گره و برای $\text{RIGHT} - \text{ARC}_d$ چپ‌ترین گره است) جایگزین می‌شوند. گذار SHIFT_p اولین گره‌ی موجود در بافر را استخراج می‌کند و آن را درون پشته می‌گذارد و آن را با برچسب اجزای سخن p برچسب می‌زند. گذار SWAP دومین بالاترین گره را از پشته استخراج می‌کند و آن را به بافر بر می‌گرداند منوط به این شرایط که دو گره‌ی بالای پشته هنوز به ترتیبی که توسط جمله داده شده بود قرار داشته باشند. به لطف گذار SWAP می‌تواند درخت‌های غیر افکنشی دلخواه را اداره کند. تنها نکته‌ای که باید بدان توجه شود این است که قبل از رسیدن به ترتیب نهایی ، هر کلمه بایستی با گذار SHIFTp به درون پشته نشانده شود، که مطمئن باشید که هر گره/کلمه در درخت خروجی برچسب خواهد خورد.

به روش‌های تجزیه‌ی وابستگی مبتنی بر گذار اصطلاحا تجزیه‌ی وابستگی جابه‌جایی-کاهش[۳۰] گفته می‌شود.

Transition		Condition
LEFT-ARC$_d$	$([\sigma\|i,j],B,A,\pi,\delta) \Rightarrow ([\sigma\|j],B,A\cup\{(j,i)\},\pi,\delta[(j,i)\rightarrow d])$	$i \neq 0$
RIGHT-ARC$_d$	$([\sigma\|i,j],B,A,\pi,\delta) \Rightarrow ([\sigma\|i],B,A\cup\{(i,j)\},\pi,\delta[(i,j)\rightarrow d])$	
SHIFT$_p$	$(\sigma,[i\|\beta],A,\pi,\delta) \Rightarrow ([\sigma\|i],\beta,A,\pi[i\rightarrow p],\delta)$	
SWAP	$([\sigma\|i,j],\beta,A,\pi,\delta) \Rightarrow ([\sigma\|j],[i\|\beta],A,\pi,\delta)$	$0 < i < j$

شکل ۵- گذارها برای تجزیه‌ی وابستگی و برچسب‌زنی توأم، گسترش یافته‌ی سیستم پشته‌ی $\sum$[12]. به عنوان لیستی ارائه شده است که سر آن در سمت راست است (و ته آن σ) بافر B به عنوان لیستی که سر آن در سمت چپ (و ته آن σ) نگارش $f[a \rightarrow b]$ برای مشخص کردن تابعی استفاده می‌شد که دقیقا شبیه f است به جز اینکه a را به b نگاشت می‌کند.

۲-۲-۱-۱-۱-تجزیه‌ی جابه‌جایی-کاهش

در مرجع [۲] آمده است: الگوریتم‌های تجزیه‌ی وابستگی جابه‌جایی-کاهش به صورت افزایشی یک جمله‌ی ورودی را از چپ به راست پردازش می‌کنند. در چارچوبی که به عنوان "arc-standard" شناخته شده است، تجزیه‌گر یکی از سه اقدام زیر را در هر گام اجرا می‌کند:

- SHIFT (SH): جابه‌جایی اولین کلمه در صف، q_0، بر روی پشته
- REDUCE-RIGHT (RR): دو درخت بالای روی پشته (s_0, s_1) را در یک زیردرخت $s_0 \curvearrowleft s_1$ ترکیب می‌کند.
- REDUCE-LEFT (RL): دو درخت بالای روی پشته (s_0, s_1) را در یک زیردرخت $s_0 \curvearrowright s_1$ ترکیب می‌کند.
- که S=(...,s_1, s_0) پشته‌ای از درخت‌هاست و صف ورودی به صورت زیر است :

$$Q=(q_0, q_1, \dots, q_{n-j-1})=(w_j, w_{j+1}, \dots, w_{n-1})$$

j اندیس اولین کلمه در صف Q است و n تعداد کلمات در جمله‌ی ورودی است. توجه داشته باشید که $s_0 \curvearrowright s_1$ یک درخت ترکیبی را مشخص می‌کند که s_1 فرزند s_0 است. برای مقابله با درگیری بین بیش از یکی از این اقدام‌ها، هر اقدام با یک امتیازی مرتبط است و امتیاز یک حالت تجزیه‌گر امتیاز کل

[30] Shift-reduce dependency parse

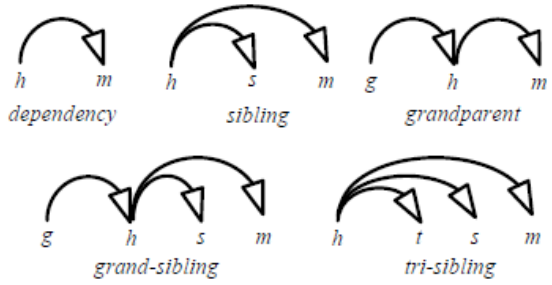
اقدام‌هایی است که اعمال شده است. برای آموزش مدل، الگوریتم پرسپترون میانگین با به‌روزرسانی اولیه در نظر گرفته می‌شود. با به‌روز رسانی اولیه هر زمان که دنباله‌ی اقدام طلایی از پرتو بیرون قرار بگیرد، پارامترها سریعا با باقی جمله که نادیده گرفته شده است به‌روز رسانی می‌شود.

۲-۲-۱-۲-تجزیه‌ی وابستگی مبتنی بر گراف

در مرجع [۱] آمده است: تجزیه‌ی وابستگی مبتنی بر گراف مسئله را به صورت یافتن درخت با بیشترین امتیاز از گراف جهت‌دار بررسی می‌کند. براساس رمزگشایی، برنامه‌نویسی پویا می‌تواند کارایی درخت بهینه را در فضای جستجوی بزرگ پیدا کند. در مدل مبتنی بر گراف امتیاز درخت وابستگی در امتیازهای بخش‌های کوچک (زیر درخت‌ها) در نظر گرفته شده است.

$$Score_{syn}(x,t,d) = w_{syn}.f_{syn}(x,t,d) = \sum_{p \subseteq d} Score_{syn}(x,t,p)$$

که p یک بخش امتیازدهی است که شامل یک یا تعداد بیشتری وابستگی در درخت وابستگی d است. شکل ٦ انواع متفاوتی از بخش‌های امتیازدهی را نشان می‌دهد که در مدل‌های مبتنی بر گراف فعلی استفاده شده است.



شکل ٦- انواع متفاوتی از بخش‌های امتیازدهی استفاده شده در مدل‌های مبتنی بر گراف فعلی[۱۱]

[۱۳] یک الگوریتم رمزگشایی $O(n^3)$ برای تجزیه‌ی وابستگی پیشنهاد داده است. براساس این الگوریتم، [١٤] مدل مرتبه اول را پیشنهاد می‌کند که بخش‌های امتیازدهی تنها شامل وابستگی‌ها هستند. مدل مرتبه‌ی دوم [15] بخش‌های sibling را ترکیب می‌کند و به زمان تجزیه‌ی $O(n^3)$ احتیاج دارد. مدل مرتبه‌ی دوم [١٦] بخش‌های sibling و پدربزرگ را نیز ترکیب می‌کند و به زمان تجزیه‌ی $O(n^4)$ نیاز دارد. گرچه، بخش‌های پدربزرگ به آن‌هایی محدود شده است که از خارجی‌ترین نوه‌ها تشکیل شده است. [۱۷] الگوریتم‌های رمزگشایی کارایی برای مدل‌های مرتبه سوم از $O(n^4)$ ارائه دادند.
در فصل بعد، پیاده‌سازی دو نسخه از مدل‌های مرتبه ۳ (مدل ۱ و مدل ۲) شرح داده شده است.

مدل ۱ تنها grand-sibling را ترکیب می‌کند گرچه مدل ۲ هر دو بخش‌های grand-sibling و tri-sibling را ترکیب می‌کند. از سه نسخه از مدل‌های تجزیه‌ی وابستگی مبتنی بر گراف استفاده می‌شود.

- مدل مرتبه اول(۰۱)
- مدل مرتبه دوم(۰۲): بدون استفاده از ویژگی‌های grand-sibling[۳۱]
- مدل مرتبه سوم(۰۳)

از مدل‌های خطی برای تعریف امتیاز درخت وابستگی استفاده می‌شود. برای مدل مرتبه‌ی سوم، امتیاز درخت وابستگی به صورت زیر نشان داده شده است:

$$\begin{aligned} Score_{syn}(x,t,d) &= \sum_{\{(h,m)\}\subseteq d} w_{dep} \cdot f_{dep}(x,t,h,m) \\ &+ \sum_{\{(h,s)(h,m)\}\subseteq d} w_{sib} \cdot f_{sib}(x,t,h,s,m) \\ &+ \sum_{\{(g,h)(h,m)\}\subseteq d} w_{grd} \cdot f_{grd}(x,t,g,h,m) \\ &+ \sum_{\{(g,h)(h,s)(h,m)\}\subseteq d} w_{gsib} \cdot f_{gsib}(x,t,g,h,s,m) \end{aligned}$$

برای مدل‌های مرتبه اول و مرتبه دوم، فرمول بالا با غیرفعال کردن بخش‌های اضافی مورد ویرایش قرار می‌گیرد. برای ویژگی‌های تجزیه، از یک عمل استاندارد برای تجزیه‌ی وابستگی مبتنی بر گراف استفاده می‌شود، زیرا این ویژگی‌ها به میزان زیادی با الگوریتم‌های رمزگشایی توأم ارتباط دارند، ویژگی‌ها به صورت زیر است:

ویژگی‌های وابستگی، $f_{dep}(x,t,h,m)$:

- ویژگی‌های یونیگرام: $w_h t_h\ dir, w_m t_m\ dir$
- ویژگی‌های بایگرام: $w_h t_h w_m t_m\ dir\ dist$
- ویژگی‌های بین: $t_h t_b t_m\ dir\ dist$
- ویژگی‌های اطراف: $t_{h-1} t_h t_{h+1} t_{m-1} t_m t_{m+1}\ dir\ dist$

ویژگی‌های sibling، $f_{sib}(x,t,h,s,m)$:

[۳۱]- این مدل مرتبه‌ی ۲ ویژگی‌های پدربزرگ را که از نوه‌ها به جای نوه‌های داخلی‌تر تشکیل شده است، ترکیب می‌کند.

$$w_h t_h w_s t_s w_m t_m \; dir$$

ویژگی‌های پدربزرگ، $f_{grd}(x,t,g,h,m)$

$$w_g t_g w_h t_h w_m t_m \; dir \; gdir$$

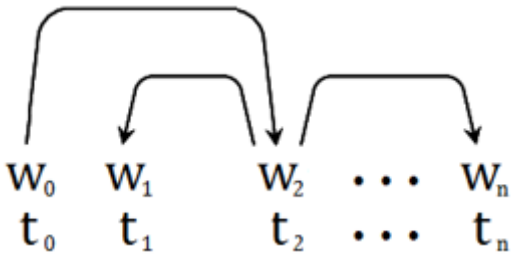
ویژگی‌های Grand-sibling، $f_{gsib}(x,t,g,h,s,m)$

$$w_g t_g w_h t_h w_s t_s w_m t_m \; dir \; gdir$$

که b اندیسی بین h و m را مشخص می‌کند؛ dir و dist جهت و فاصله‌ی (h,m) را نشان می‌دهد. Gdir جهت (g,h) را نشان می‌دهد. همچنین از ویژگی‌های رو به عقب با تعمیم از ویژگی‌های بسیار خاص بر روی شکل‌های کلمه، برچسب‌های اجزای سخن، جهت‌ها و فاصله‌ها هم استفاده می‌شود تا ویژگی‌های تنک کمتری روی برچسب‌های اجزای سخن وجود داشته باشد یا اینکه گره‌های کمتری در نظر گرفته شود. برای جلوگیری از تولید تعداد زیادی ویژگی تنک، زمانی که برچسب‌های اجزای سخن به جای دیگر نودها استفاده می‌شود حداکثر دو شکل کلمه در یک زمان در ویژگی‌های sibling ، پدربزرگ و grand-sibling قرار می‌گیرند؛ در ضمن، حداکثر چهار برچسب اجزای سخن در یک زمان برای ویژگی‌های اطراف در نظر گرفته می‌شود.

۲-۳- روش پایپلاین پایه

جمله‌ی ورودی متشکل از n واژه ($x = w_1 \dots w_n$) است و دنباله‌ی برچسب اجزای سخن متناظر با هر واژه ($t = t_1 \dots t_n$) داده شده که مقدار هر برچسب از میان مجموعه‌ی برچسب[۳۲] اجزای سخن انتخاب می‌شود ($1 \leq i \leq n,\ t_i\ \tau\epsilon$). یک درخت وابستگی با $d = \{(h,m): 0 \leq h \leq n, 0 < m \leq n\}$ مشخص می‌شود، که (h,m) یک وابستگی $w_h \rightarrow w_m$ را ارائه می‌کند که کلمه‌ی سر (یا پدر) w_h است و پیرایندهٔ[۳۳](یا فرزند) w_m است. w_0 یک نشانه‌ی ریشه‌ی مصنوعی است که برای ساده‌سازی صورت مسئله استفاده شده است [۱].



شکل ۷- جمله‌ی ورودی و دنباله‌ی برچسب

[32] tagset
[33] -modifier

روش پایپلاین با برچسب‌زنی اجزای سخن و تجزیه‌ی وابستگی به صورت دو مسئله‌ی آبشاری برخورد می‌کند. در ابتدا، دنباله‌ی برچسب اجزای سخن بهینه $\hat{t}$ به صورت زیر مشخص شده است.

$$\hat{t} = arg\max_{t} Score_{pos}(x, t)$$

سپس، درخت وابستگی بهینه‌ی $\hat{d}$ بر اساس x و $\hat{t}$ به صورت زیر مشخص شده است.

$$\hat{d} = arg\max_{d} Score_{syn}(x, \hat{t}, d)$$

۲-٤- ضرورت استفاده از مدل‌های توأم برای برچسب‌زنی و تجزیه‌ی وابستگی

در تجزیه‌ی وابستگی، ویژگی‌ها شامل برچسب‌های اجزای سخن بسیار موثرند، زیرا ویژگی‌های واژگانی[۳٤] خالص منجر به مسئله‌ی جدی تنکی داده‌ها می‌شود [۱]. در حال حاضر دو وظیفه‌ی برچسب‌زنی و تجزیه‌ی وابستگی در بسیاری از زبان‌ها به صورت مستقل مورد بررسی قرار می‌گیرد و معمولا، برچسب‌زنی اجزای سخن و تجزیه‌ی وابستگی به روش پایپلاین مدل می‌شوند. به دلیل بهره نبردن اجزای سخن از اطلاعات نحوی، این وظیفه با دقت کمتری انجام می‌گیرد و همچنین برچسب‌های دقیق اجزای سخن می‌توانند در انجام تجزیه‌ی وابستگی با دقت بیشتر کمک کنند. به عبارت دیگر در انجام مستقل این دو وظیفه انتشار خطا بوجود می‌آید. همچنین مدل‌های توأم رویکردی رایج و تأثیرگذار برای انجام همزمان وظایف مشابه هستند (به عنوان مثال برای ارائه مدل توأم از ریشه‌یابی[۳۵] و برچسب‌زنی اجزای سخن، شناسایی موجودیت‌های اسمی[۳٦] و تجزیه، برچسب‌زنی نقش‌های مفهومی[۳۷] و تجزیه، تجزیه و ترجمه ماشینی[۳۸] و غیره)[۱]. گرچه دقت تجزیه‌ی وابستگی زمانی که از داده‌های برچسب‌خورده استفاده می‌شود خوب است اما اگر بخواهیم تجزیه‌ی وابستگی را روی یک متن جدید اعمال کنیم چون تعداد زیادی واژگان جدید دارد دقت تجزیه بسیار افت می‌کند ولی اگر از مدل‌های توأم بهره ببریم می‌توان از اطلاعات نحوی برای برچسب‌زنی استفاده کرد و آن را بهبود داد. و اگر دقت برچسب‌زنی بهتر شود تجزیه‌ی وابستگی نیز بهبود پیدا می‌کند و این افت دقت شدید نخواهد بود.

---

[34] lexical
[35] Lemmatization
[36] Named entity recognition
[37] Semantic role labling
[38] MT: Machine translation

۲-٤- نتیجه‌گیری

در این فصل ابتدا مقدماتی در مورد برچسب‌زنی اجزای سخن و روش‌های انجام آن و نیز روش‌های حل مسئله‌ی تجزیه‌ی وابستگی ارائه شد و همچنین روش پایپلاین برچسب‌زنی و تجزیه‌ی وابستگی را توضیح دادیم. در انتهای فصل در مورد ضرورت استفاده از مدل‌های توأم بحث کردیم و دریافتیم که برای تجزیه‌ی وابستگی یک متن خام با دقت خوب ناگزیریم که از برچسب‌زنی و تجزیه‌ی وابستگی به صورت توأم بهره ببریم.

فصل سوم

روش‌های توأم برچسب‌زنی اجزای سخن و تجزیه‌ی وابستگی

۳-روش‌های توأم برچسب‌زنی اجزای سخن و تجزیه‌ی وابستگی

مقدمه

برچسب‌زنی اجزای سخن و تجزیه‌ی وابستگی، دو وظیفه‌ی اساسی در پردازش زبان طبیعی است. معمولا، برچسب‌زنی یک گام پیش‌پردازش برای تجزیه‌ی وابستگی به‌ویژه در معماری پایپلاین است. همانطور که در فصل قبل اشاره کردیم، دو مسئله‌ی اصلی در سیستم‌های پایپلاین وجود دارد:۱- تجزیه‌ی وابستگی از مشکل انتشار خطا رنج می‌برد؛ ۲- برچسب‌زنی اجزای سخن از اطلاعات نحوی مهم و مفید برای ابهام‌زدایی بهره‌برداری نمی‌کند [۱۱]. البته دقت تجزیه‌ی وابستگی بر روی داده‌های برچسب‌خورده خوب است و لزوم اصلی رفتن به سمت مدل‌های توأم استفاده در کاربرد واقعی تجزیه‌ی وابستگی متن خام است. از طرف دیگر کاربرد اصلی روش‌های توأم در تطبیق دامنه است یعنی کاربرد تجزیه‌گر بر روی دامنه‌ای دیگر که بر روی آن آموزش ندیده است. و بایستی دقت به دلیل واژگان جدید و دیده به میزان زیادی افت نکند.

مدل‌های توأم به سه دسته‌ی مبتنی بر گذار، مبتنی بر گراف و روش‌های پشته‌ای تقسیم می‌شوند که روش‌های پشته‌ای دو تجزیه‌گر مبتنی بر گذار و مبتنی بر گراف را با هم ترکیب می‌کنند.

۳-۱- مدل‌های توأم

در روش توأم، هدف این است که به طور همزمان دو مسئله حل شود.

$$(\hat{t}, \hat{d}) = arg \max_{t,d} Score_{joint}(x, t, d)$$

تحت مدل خطی، امتیاز درخت وابستگی برچسب خورده به صورت زیر است:

$$Score_{joint}(x, t, d) = Score_{pos}(x, t) + Score_{syn}(x, t, d)$$
$$= w_{pos \oplus \text{syn}} \cdot f_{pos \oplus \text{syn}}(x, t, d)$$

که $f_{pos \oplus \text{syn}}(.)$ به معنای الحاق $f_{pos}(.)$ و $f_{syn}(.)$ است. تحت مدل توأم وزن‌های ویژگی‌های اجزای سخن و ویژگی‌های نحوی، $w_{pos \oplus \text{syn}}$، به صورت همزمان یاد گرفته می‌شود. انتظار می‌رود که ویژگی‌های اجزای سخن و ویژگی‌های نحوی بتوانند با یکدیگر تعامل کنند تا یک نتیجه‌ی توأم بهینه را مشخص کند[۱].

۳-۲- مدل‌های توأم مبتنی بر گذار

تجزیه‌گرهای مبتنی بر گذار پیشین از استنتاج حریصانه‌ی اول-بهترین[۳۹] استفاده می‌کردند و رده‌بندها را به صورت محلی آموزش می‌دادند، ولی کارهای اخیر نشان داده‌اند که با استفاده از جستجوی پرتو و یادگیری ساختاری سراسری دقت بیشتری برای کاهش خطای انتشار به‌دست می‌دهند. به خصوص، به نظر می‌رسد که مدل‌های آموزش داده شده‌ی سراسری[۴۰] می‌توانند فضای ویژگی بسیار غنی‌تری را به نسبت رده‌بندهای آموزش دیده‌ی محلی استخراج کنند، چون تجزیه و برچسب‌زنی توأم اندازه‌ی فضای جستجو را افزایش می‌دهد احتمال دارد تا به ویژگی‌های جدیدی نیز نیاز داشته باشد [۱۰]. بنابراین یکی از چالش‌ها در مدل‌های توأم کافی و جامع نبودن ویژگی‌های درنظر گرفته شده است در همین راستا بایستی برای حل این مشکل ویژگی‌های کارا و بهینه را استخراج کرد.

۳-۲-۱- یک مدل توأم مبتنی برگذار با درخت غیرافکنشی برچسب‌دار

در این بخش الگوریتم استنتاج و یادگیری یک مدل توأم مبتنی بر گذار ارائه شده در [۱۰] توضیح داده شده و سپس ویژگی‌های به کار رفته در این مدل مورد بررسی قرار گرفته است و چون جستجو به صورت سراسری انجام گرفته ویژگی‌های جدیدی برای این مدل در نظر گرفته شده است. درخت وابستگی حاصل در این مدل یک درخت غیرافکنشی برچسب‌دار است.

۳-۲-۱-۱- استنتاج و یادگیری در مدل توأم

الگوریتم جستجوی پرتو برای نتیجه گرفتن بهترین تجزیه‌ی y برای جمله‌ی x در شکل ۸ خلاصه شده است. به‌علاوه برای جمله‌ی x یک بردار وزن w ورودی نظیر یک مدل خطی برای امتیازدهی گذارها بدون ترتیب‌ها و دو پارامتر هرس b_1 و b_2 را در نظر می‌گیرد. یک فرضیه‌ی تجزیه‌ی h با ترتیب h.c، یک امتیاز h.s و بردار ویژگی h.f نمایش داده می‌شود. فرضیه‌ها در لیست BEAM نگه‌داری می‌شوند، که با امتیازهای نزولی مرتب‌سازی می‌شود و فرضیه‌ی h_0 متناظر با ترتیب اولیه‌ی $c_s(x)$ با امتیاز صفر مقداردهی اولیه می‌شود و تمام ویژگی‌ها را برابر صفر قرار می‌دهد (خطوط ۱-٤). در حلقه‌ی اصلی(خطوط ۵-۱۳) یک مجموعه از فرضیه‌های جدید منتج و در لیست TMP نگهداری شده است، که در نهایت هرس شده‌اند و به عنوان مقدار جدید BEAM نسبت داده شده است. حلقه‌ی اصلی زمانی

[39] best-first
[40] globally learned models

خاتمه پیدا می‌کند که همه‌ی فرضیه‌ها در BEAM شامل ترتیب نهایی باشند، و درخت وابستگی از فرضیه‌ای که بالاترین امتیاز را برگردانده است استخراج شده باشد(خطوط ١٤-١٦) [١٠].

```
PARSE(x, w, b1, b2)
 1  h0.c ← cs(x)
 2  h0.s ← 0.0
 3  h0.f ← {0.0}^dim(w)
 4  BEAM ← [h0]
 5  while ∃h ∈ BEAM : h.c ∉ Ct
 6     TMP ← [ ]
 7     foreach h ∈ BEAM
 8        foreach t ∈ T : PERMISSIBLE(h.c, t)
 9           h.f ← h.f + f(h.c, t)
10           h.s ← h.s + f(h.c, t) · w
11           h.c ← t(h.c)
12           TMP ← INSERT(h, TMP)
13     BEAM ← PRUNE(TMP, b1, b2)
14  h ← TOP(BEAM)
15  y ← TREE(x, h.c)
16  return y
```

شکل ٨- الگوریتم جستجوی پرتو برای تجزیه‌ی وابستگی و برچسب‌زنی توأم جمله‌ی ورودی x با بردار وزن‌دار w و پارامترهای beam b_1, b_2 . نمادهای h.c و h.s و h.f به ترتیب مشخص‌کننده‌ی ترتیب، امتیاز و نمایش ویژگی فرضیه‌ی h اند. H.c.A مجموعه‌ی کمان h.c را نشان می‌دهد.

مجموعه‌ی فرضیه‌های جدید در دو حلقه‌ی درونی ایجاد می‌شود(خطوط ٧-١٢)، که هر فرضیه‌ی h در BEAM با استفاده از هر گذار قابل قبول t برای هر ترتیب h.c به‌روز رسانی می‌شود. نمایش ویژگی فرضیه‌ی جدید با اضافه کردن بردار ویژگی f(t,h.c) برای زوج ترتیب-گذار جاری به بردار ویژگی فرضیه‌ی قدیمی به‌دست می‌آید(خط ٩). به طور مشابه، امتیاز فرضیه‌ی جدید مجموع امتیاز f(t,h.c).w زوج ترتیب-گذار جاری و امتیاز فرضیه‌ی قبلی است(خط ١٠). نمایش ویژگی/امتیاز یک تجزیه‌ی کامل y برای x با دنباله‌ی گذار $C_{0,m}$ است پس مجموع نمایش‌های ویژگی/امتیازهای زوج‌های ترتیب-گذار در $C_{0,m}$ برابر است با:

$$f(x,y) = \sum_{(c,t)\epsilon C_{0,m}} f(c,t)$$

$$s(x,y) = \sum_{(c,t)\epsilon C_{0,m}} f(c,t) \,.\, w$$

در نهایت ترتیب فرضیه‌ی جدید با ارزیابی t(h.c) به‌دست می‌آید (خط ۱۱). سپس فرضیه‌ی جدید به TMP در مرتبه‌ی مرتب‌سازی امتیاز افزوده می‌شود(خط ۱۲) [۱۰].

پارامترهای هرس b_2 و b_1 تعداد فرضیه‌هایی را که در پرتو مجاز هستند مشخص می‌کند و در عین حال کنترل معاوضه‌ای[41] بین ابهام نحوی و ساخت‌واژی است. در ابتدا، b_1 را از بیشترین امتیاز فرضیه‌ها با درخت‌های وابستگی متمایز استخراج می‌کنیم. سپس b_2 از بیشترین امتیاز فرضیه‌های باقیمانده استخراج می‌شود، که قبلا در روش پرتو انواع برچسب‌ها را در درخت‌های وابستگی داشته‌اند. در این روش، از پر شدن پرتو با انواع برچسب‌های یک درخت یکسان جلوگیری می‌شود که در آزمایشات مقدماتی مضر شناخته شده است. یک چیز نهایی که بایستی در مورد الگوریتم استنتاج به آن توجه شود مفهوم مجاز برای گذار t خارج از ترتیب c است که می‌تواند برای گیر نیفتادن در محدودیت‌های رسمی روی گذارها استفاده شود. این حقیقت که اجرای گذار SHIFTp با یک بافر خالی غیر ممکن است یا غیر مجاز بودن اجرای گذار LEFT-ARCd با گره‌ی ریشه‌ی خاص بالای پشته، برای فیلتر کردن برچسب‌های وابستگی نامحتمل هم مورد استفاده قرار می‌گیرد. بنابراین در آزمایش‌های بعد معمولا تجزیه کننده طوری محدود می‌شود که SHIFTp تنها زمانی قابل قبول است که p یکی از k تا بهترین برچسب‌های اجزای سخن با امتیازی باشد که بیشتر از α نیست که امتیاز α زیر امتیاز بهترین برچسب است، که توسط یک برچسب‌زن پیش‌پردازنده مشخص شده است. همچنین نمونه‌هایی از LEFT-ARC_d و RIGHT-ARC_d نیز فیلتر می‌شود، که d در داده‌های آموزشی برای ترکیب برچسب اجزای سخن پیش‌بینی شده‌ی سر و وابسته رخ نمی‌دهد. این رویه به سمت سرعت چشمگیری هدایت می‌شود [۱۰].

به منظور یادگیری بردار وزن w از مجموعه‌ی آموزشی $\{(x_j, y_j)\}_{j=1}^{\tau}$ جملاتی با درخت‌های وابستگی برچسب‌دار، یکی از انواع پرسپترون ساخت‌یافته که توسط [۹] معرفی شد مورد استفاده قرار می‌گیرد، که N تکرار را روی داده‌ی آموزشی انجام می‌دهد و بردار وزن را برای هر جمله‌ی x_j که بیشترین امتیاز تجزیه‌ی y^* با y_j متفاوت است به‌روز رسانی می‌کند.

$$W^{i+1} = W^i + \tau(f(x_j, y_j) - f(x_j, y^*))$$

که $$\tau = \frac{f(x_j,y_j)-f(x_j,y^*)}{\|f(x_j,y_j)-f(x_j,y^*)\|^2}$$

[41] Trade off

همچنین روش به‌روز رسانی پیشین نیز مورد استفاده قرار می‌گیرد، بدین معنی که در طول یادگیری، زمانی که فرضیه‌ی متناظر تجزیه‌ی طلایی y_j به خارج از پرتو برود جستجوی پرتو خاتمه پیدا می‌کند و به‌روز رسانی با توجه به دنباله‌ی گذار جزئی که تا این لحظه ساخته شده است انجام می‌گیرد. سرانجام از روش میانگین‌گیری همه‌ی بردارهای وزن استفاده می‌شود[۱۰].

۳-۲-۱-۲- نمایش‌های ویژگی

نمایش ویژگی f(x,y) یک جمله‌ی ورودی x با تجزیه‌ی y که به نمایش‌های ویژگی f(c,t) تجزیه می‌شود برای گذار t(c) احتیاج است تا y را از $c_s(x)$ نتیجه بگیرد. ویژگی‌ها می‌تواند به هر جنبه از یک ترتیب اشاره کند که در پشته‌ی $\sum$، بافر B، مجموعه‌ی کمان A و برچسب‌زنی‌های π و δ رمز شده است. به علاوه فرض می‌شود که هر کلمه w در ورودی به k نامزد برچسب‌های اجزای سخن $\pi_i(w)$ با امتیازهای نظیر $s(\pi_i(w))$ نسبت داده شود [۱۰].

Features involving word prefixes and suffixes
$\pi_i(B_0)p_2(B_0), \pi_i(B_0)s_2(B_0), \pi_i(B_0)p_1(B_0)p_1(\Sigma_0)$
$\pi_i(\Sigma_0)p_1(\Sigma_0)p_1(\Sigma_1), \pi_i(\Sigma_0)s_1(\Sigma_0)s_1(\Sigma_0)$
$\pi_i(\Sigma_0)p_2(\Sigma_0)s_3(\Sigma_1), \pi_i(\Sigma_0)s_3(\Sigma_0)p_2(\Sigma_1)$
$\pi_i(\Sigma_0)w(B_0)s_1(\Sigma_0), \pi_i(\Sigma_0)w(B_0)s_2(\Sigma_0)$
Features involving tag score differences and ranks
$\pi_i(B_0)[s(\pi_1(B_0)) - s(\pi_i(B_0))]$
$\pi_i(B_0)\pi_i(\Sigma_0)[s(\pi_1(B_0)) - s(\pi_i(B_0))]\ i$
$\pi_i(B_0)[s(\pi_1(B_0)) - s(\pi_i(B_0))]\pi(\Sigma_0)$
$w(B_0)[s(\pi_1(B_0)) - s(\pi_i(B_0))]\pi(\Sigma_0)$

شکل ۹- الگوهای اختصاص داده شده برای برچسب‌زنی. از B_i و $\sum_i$ استفاده می‌شود تا نشانه‌ی iام به ترتیب در پشته‌ی $\sum$ و بافر B با شروع اندیس‌گذاری از 0 مشخص شود، و از اجرا کننده‌های زیر برای استخراج ویژگی‌های یک نشانه استفاده می‌شود.i امین بهترین برچسب $= (\ )\pi_i$؛ امتیاز i امین بهترین برچسب $= s(\pi_i)$ ؛ برچسب پیش‌بینی شده‌ی نهایی= $\pi()$؛ فرم کلمه =w()؛کلمه‌ی پیشوند iتا کاراکتر $= (\ )p_i$؛ کلمه‌ی پسوند iتا کاراکتر $= (\ )s_i$. تفاوت امتیازها در گام‌های گسسته‌ی ۰,۰۵در نظر گرفته شده است [۱۰].

ویژگی‌هایی استفاده شده در سیستم از [۱۸] گرفته شده است، البته با دو تفاوت مهم. اول از همه، تمام ویژگی‌هایی که یک ترتیب تجزیه‌ی arc-eager را پیش‌فرض قرار می‌دهند حذف می‌کند، زیرا سیستم گذار مد نظر یک ترتیب arc-standard را تعریف می‌کند. دوما، هر ویژگی‌ای که به برچسب اجزای

سخن یک کلمه‌ی w در بافر B سیستم اشاره کند، به جای برچسب پیش‌بینی شده‌ی نهایی به برچسب بالاترین امتیاز $\pi_1(w)$ اشاره می‌کند. در مقابل، برای یک کلمه در پشته‌ی $\sum$، ویژگی‌های برچسب اجزای سخن به برچسب $\pi(w)$ ای اشاره می‌کند که با جابه‌جایی w در پشته انتخاب شده است(که ممکن است با $\pi_1(w)$ یکسان باشد یا نباشد)[۱۰].

علاوه بر ویژگی‌های استاندارد برای تجزیه‌ی وابستگی مبتنی بر گذار، ویژگی‌های مخصوصی برای بهبود گام برچسب‌زنی در مدل توأم افزوده شده است. الگوها برای این ویژگی‌ها که در شکل ۹ نشان داده شده‌اند، iامین بهترین برچسب نسبت داده شده به اولین کلمه‌ی بافر B (کلمه‌ی بعدی بایستی با گذار SHIFTp جابه‌جا شود) را در ترکیب با کلمه‌های همسایگی، پیشوندهای کلمه، پسوندهای کلمه، تفاوت‌های امتیاز و رتبه‌ی برچسب درگیر می‌کنند. و همچنین از دو مجموعه‌ی ویژگی اضافی استفاده می‌شود، که به آن‌ها به ترتیب ویژگی‌های گراف (G) و ویژگی‌های خوشه (C) گفته می‌شود.

از یک کرنل Hash برای نگاشت ویژگی‌ها به وزن‌ها استفاده می‌شود. بیشترین زمان محاسبه در تجزیه‌گرهای غنی از نظر ویژگی در بازیابی اندیس هر ویژگی در بردار وزن صرف می‌شود. این کار معمولا با یک جدول hash انجام می‌گیرد، افزایش سرعت چشمگیری با استفاده از یک کرنل hash صورت می‌گیرد که به‌سادگی جدول مراجعه را با یک تابع Hash جایگزین می‌کند.

۳-۲-۲- مدل توأم مبتنی بر گذار افزایشی

در این بخش اولین رویکرد افزایشی که برای وظیفه‌ی برچسب‌زنی اجزای سخن و تجزیه‌ی وابستگی پیشنهاد شده است مورد بررسی قرار می‌گیرد. با توجه به جمله‌ی تقسیم شده، مدل به صورت همزمان، برچسب‌های اجزای سخن ممکن و روابط وابستگی را با جستجوی پرتو داده شده در نظر می‌گیرد و بهترین تجزیه را همراه با برچسب‌های اجزای سخن به عنوان خروجی می‌دهد. گرچه مدل ترکیبی دو چالش به وجود می‌آورد. اولا اینکه چون فضای جستجوی ترکیبی بزرگ است، رمزگشایی کارا دشوار است در حالی که استفاده ساده از پرتو به احتمال زیاد کیفیت جستجو را تنزل می‌دهد. دوما، چون مدل پیشنهادی برچسب‌زنی اجزای سخن و تجزیه‌ی وابستگی توأم در روش چپ به راست پیشنهاد شده

است، مدل نمی‌تواند برچسب‌های اجزای سخن پیش‌بینی[۴۲] را برای مشخص کردن اقدام بعدی مورد بهره‌برداری قرار دهد[۲].

برای کار با این فضای جستجوی بزرگ بایستی برنامه‌نویسی پویایی برای تجزیه‌ی جابه‌جایی-کاهش در نظر گرفته می‌شود که مدل را قادر سازد تا حالت‌های تجزیه‌ی معادل را بسته‌بندی کرده در نتیجه سرعت و دقت را بهبود بخشد. برای غلبه بر فقدان اطلاعات برچسب‌های اجزای سخن پیش‌بینی مفهوم ویژگی‌های تأخیری معرفی می‌شود. ویژگی‌های تأخیری[۴۳] ویژگی‌هایی می‌باشند که شامل برچسب‌های اجزای سخن ویژه‌اند و باید در گامی که برچسب‌های پیش‌بینی مشخص می‌شوند ارزیابی شوند. این مدل توأم بر روی هر زبانی قابل اجراست که برای یک تجزیه‌گر جابه‌جایی-کاهش افکنشی خوب عمل می‌کند[۲].

۳-۲-۲-۱- برچسب‌های اجزای سخن baseline

در این مدل از یک برچسب‌زن اجزای سخن baseline استفاده شده است. لیست ویژگی‌هایی که مورد استفاده قرار گرفته در جدول ۱ نشان داده شده است. این مدل با پرسپترون میانگین[۴۴] آموزش داده شده است و رمزگشایی آن با استفاده از الگوریتم ویتربی با جستجوی پرتو انجام گرفته است[۲].

در این کار از یک واژه‌نامه‌ی برچسب و برچسب‌های مجموعه‌ی بسته[۴۵] استفاده شده است که منجر به بهبودهایی هم در سرعت و هم در دقت می‌شود. در طول آموزش، مدل همه‌ی کلمه-برچسب‌های زوج را در یک واژه‌نامه برچسب ذخیره می‌کند، و برای هر کلمه‌ای که بیشتر از N بار در مجموعه‌ی آموزشی رخ می‌دهد، رمزگشا یکی از برچسب‌ها را نسبت می‌دهد که در داده‌ی آموزشی دیده شده است. برای کلماتی که در واژه‌نامه وجود ندارند، هر برچسب ممکنی را در نظر می‌گیرد. همچنین یک واژه‌نامه برای برچسب‌های مجموعه‌ی بسته درست شده است و به رمزگشا اجازه داده می‌شود تا این برچسب‌ها را به کلماتی که در واژه‌نامه لیست شده است نسبت دهد[۲].

---

[42] look-ahead
[43] delayed features
[44] averaged perceptron
[45] closed set

۳-۲-۲-۲-تجزیه‌گرهای baseline

برای تجزیه‌گرهای baseline دو تجزیه‌گر وابستگی ساخته می‌شود: یکی پیاده‌سازی دوباره‌ی تجزیه‌گر [۱۹] (از این به بعد Parser-HS) انجام می‌گیرد که یک تجزیه‌گر وابستگی جابه‌جایی-کاهش است که با برنامه‌نویسی پویا با استفاده از پشته‌ی با ساختار گرافی ایجاد می‌شود. و دومی گسترشی از Parser-HS با ترکیب مجموعه‌ی غنی‌شده‌ی ویژگی‌های گرفته شده از [۱۸] است (از این به بعد Parser-ZN) که در اصل یک تجزیه‌ی وابستگی Arc-eager بدون برنامه‌نویسی پویاست [۲].

۳-۲-۲-۳- ادغام حالت‌های معادل

در مرجع [۲] برنامه‌نویسی پویا حالت‌های معادل را ادغام می‌کند: اگر دو حالت بردار ویژگی مشابهی را تولید کنند، آن‌ها در یک حالت ادغام می‌شوند. به طور رسمی، یک حالت تجزیه‌گر (یا پیکربندی) ψ با $\langle l,i,j,S\rangle$ توضیح داده می‌شود ،که l گام فعلی است، [i...j]، span درخت بالای s_0 در پشته $S=(s_{d-1},\dots,s_0)$ است که d عمق پشته است. معادل بودن دو حالت $\psi:\langle l,i,j,S\rangle$ و $\psi':\langle l',i',j',S'\rangle$ به صورت زیر تعریف می‌شود:

$$\psi\sim\psi' \; iff \quad j=j' \wedge \vec{f}(j,S)=\vec{f}(j',S'), \qquad (1)$$

که $\vec{f}(j,S)$ بردار ویژگی حالت $\langle l,i,j,S\rangle$ است. در عمل، یک مجموعه کمینه از ویژگی‌ها که ویژگی‌های کرنل $\tilde{\vec{f}}(j,S)$ نامیده می‌شود برای ارزیابی حالت‌های معادل کافی است:

$$\tilde{\vec{f}}(j,S)=\tilde{\vec{f}}(j',S') \Longrightarrow \langle l,i,j,S\rangle\sim\langle l',i',j',S'\rangle \qquad (2)$$

با ادغام حالت‌های معادل براساس این شرط، تنها نیاز است تا اطلاعات مربوط را از d درخت بالای بر روی پشته بدست آورده تا امتیاز اقدام‌های بعدی ارزیابی شود. گرچه چون زمانی که اقدام REDUCE-LEFT/RIGHT اعمال می‌شود پشته کوچک می‌شود، اغلب احتیاج است تا جزء آخر پشته از سابقه بازیابی شود. از مفهوم حالت‌های پیش‌گویی کننده‌ی $\prod(\psi)$ برای بازیابی پیوندهایی با چندین سابقه‌ی متفاوت استفاده می‌شود.

w_j	t_{j-1}	$t_{j-1} \circ t_{j-2}$	w_{j+1} [1)]
$w_j \circ E(w_{j-1})$ [2)]		$w_j \circ B(w_{j+1})$ [2)]	
$E(w_{j-1}) \circ w_j \circ B(w_{j+1})$ [3)]			
$B(w_j)$	$E(w_j)$	$P(B(w_j))$	$P(E(w_j))$
$C_n(w_j)$	$(n \in \{2, \ldots, len(w_j) - 1\})$		
$B(w_j) \circ C_n(w_j)$	$(n \in \{2, \ldots, len(w_j)\})$		
$E(w_j) \circ C_n(w_j)$	$(n \in \{1, \ldots, len(w_j) - 1\})$		
$C_n(w_j)$	*(if* $C_n(w_j)$ *equals to* $C_{n+1}(w_j)$*)*		

1) if $len(w_{j+1}) < 3$; 2) if $len(w_j) < 3$; 3) if $len(w_j) = 1$.

جدول ۱- الگوهای ویژگی برای برچسب‌زن اجزای سخن baseline، که هر t_i برچسبی است که به کلمه‌ی iام w_i نسبت داده شده است و B(w) و E(w) کاراکتر شروع و پایان w است و $C_n(w)$ nامین کاراکتر w است، و P(c) مجموعه‌ای از برچسب‌های مرتبط با کلمه‌ی تک کاراکتر c براساس واژه نامه است [۲].

۳-۲-۲-٤- ویژگی‌ها

الگوهای ویژگی که در تجزیه‌گر baseline، Parser-HS استفاده شده است در جدول ۲(a) لیست شده است، که s.w و s.t فرم و برچسب کلمه‌ی ریشه‌ی درخت s است و s.rc و s.lc راست‌ترین و چپ‌ترین بچه‌ها هستند و o بیانگر ارتباط ویژگی‌هاست. توجه داشته باشید که این ویژگی‌ها می‌تواند تنها با ۱۳ ویژگی کرنل که در جدول ۲(c) لیست شده است ساخته شود. Parser-ZN علاوه بر این، از ویژگی‌های جدول ۲(b) استفاده می‌کند که d فاصله‌ی بین گره‌های ریشه‌ی s_0 و s_1 را مشخص می‌کند، $s.v_r$ و $s.v_l$ تعداد فرزندان راست و چپ s هستند، $s.rc_2$ و $s.lc_2$ دومین راست‌ترین و دومین چپ‌ترین فرزندان s هستند. [۲].

(a)	$s_0.w$	$s_0.t$	$s_0.w \circ s_0.t$		
	$s_1.w$	$s_1.t$	$s_1.w \circ s_1.t$		
	$q_0.w$	$q_0.t$	$q_0.w \circ q_0.t$		
	$s_0.w \circ s_1.w$		$s_0.t \circ s_1.t$		
	$s_0.t \circ q_0.t$		$s_0.w \circ s_0.t \circ s_1.t$		
	$s_0.t \circ s_1.w \circ s_1.t$		$s_0.w \circ s_1.w \circ s_1.t$		
	$s_0.w \circ s_0.t \circ s_1.w$		$s_0.w \circ s_0.t \circ s_1.w \circ s_1.t$		
	$s_0.t \circ q_0.t \circ q_1.t$		$s_1.t \circ s_0.t \circ q_0.t$		
	$s_0.w \circ q_0.t \circ q_1.t$		$s_1.t \circ s_0.w \circ q_0.t$		
	$s_1.t \circ s_1.\mathrm{rc}.t \circ s_0.t$		$s_1.t \circ s_1.\mathrm{lc}.t \circ s_0.t$		
	$s_1.t \circ s_1.\mathrm{rc}.t \circ s_0.w$		$s_1.t \circ s_1.\mathrm{lc}.t \circ s_0.w$		
	$s_1.t \circ s_0.t \circ s_0.\mathrm{rc}.t$		$s_1.t \circ s_0.w \circ s_0.\mathrm{lc}.t$		
	$s_2.t \circ s_1.t \circ s_0.t$				
(b)	$s_0.w \circ d$	$s_0.t \circ d$	$s_1.w \circ d$	$s_1.w \circ d$	
	$s_0.w \circ s_0.v_l$		$s_0.t \circ s_0.v_l$		
	$s_1.w \circ s_1.v_r$		$s_1.t \circ s_1.v_r$		
	$s_1.w \circ s_1.v_l$		$s_1.t \circ s_1.v_l$		
	$s_0.\mathrm{lc}.w$	$s_0.\mathrm{lc}.t$	$s_1.\mathrm{rc}.w$	$s_1.\mathrm{rc}.t$	
	$s_1.\mathrm{lc}.w$	$s_1.\mathrm{lc}.t$	$s_0.\mathrm{lc}_2.w$	$s_0.\mathrm{lc}_2.t$	
	$s_1.\mathrm{rc}_2.w$	$s_1.\mathrm{rc}_2.t$	$s_1.\mathrm{lc}_2.w$	$s_1.\mathrm{lc}_2.t$	
	$s_0.t \circ s_0.\mathrm{lc}.t \circ s_0.\mathrm{lc}_2.t$		$s_1.t \circ s_1.\mathrm{rc}.t \circ s_1.\mathrm{rc}_2.t$		
	$s_1.t \circ s_1.\mathrm{lc}.t \circ s_1.\mathrm{lc}_2.t$				
(c)	j	$s_2.t$	$q_0.w$	$q_0.t$	$q_1.t$
	$s_1.w$	$s_1.t$	$s_1.\mathrm{rc}.t$	$s_1.\mathrm{lc}.t$	
	$s_0.w$	$s_0.t$	$s_0.\mathrm{rc}.t$	$s_0.\mathrm{lc}.t$	
(d)	d	$s_0.v_l$	$s_1.v_l$	$s_1.v_r$	
	$s_0.\mathrm{lc}.w$	$s_1.\mathrm{rc}.w$	$s_1.\mathrm{lc}.w$		
	$s_0.\mathrm{lc}_2.w$	$s_1.\mathrm{rc}_2.w$	$s_1.\mathrm{lc}_2.w$		
	$s_0.\mathrm{lc}_2.t$	$s_1.\mathrm{rc}_2.t$	$s_1.\mathrm{lc}_2.t$		

جدول ۲-(a) الگوهای ویژگی برای Parser-HS (b) الگوهای ویژگی اضافی برای Parser-ZN (c) ویژگی‌های کرنل برای Parser-HS (d) ویژگی‌های کرنل اضافی برای Parser-ZN [۲].

۳-۲-۲-۵- جستجوی پرتو با برنامه‌نویسی پویا

در تجزیه‌ی جابه‌جایی-کاهش با برنامه‌نویسی پویا، نمی‌توان به سادگی جستجوی پرتو را به عنوان یک تجزیه‌ی جابه‌جایی-کاهش بدون برنامه‌نویسی پویا اعمال کرد زیرا، هر حالت بیش از یک امتیاز منحصر بفرد ندارد. برای تصمیم‌گیری در مورد مرتب کردن حالت‌ها با پرتو مفهوم امتیاز پیشوند و امتیاز درونی در نظر گرفته می‌شود. امتیاز پیشوندی ξ امتیاز کل بهترین دنباله‌ی اقدام‌ها از حالت اولیه تا حالت فعلی است، در حالی که امتیاز درونی η امتیاز درخت بالای پشته است. با این امتیازها و مجموعه‌ای از حالت‌های پیش‌بینی کننده‌ی $\prod(\psi)$ حالت ψ، توضیح کامل حالت ψ شکل $\psi: \langle l, i, j, S; \xi, \eta, \Pi \rangle$

را به خود می‌گیرد. محاسبه‌ی امتیازهای پیشوندی و درونی در [۱۹] توضیح داده شده است. با استفاده از این امتیازها، مرتب کردن حالت‌ها به صورت زیر تعریف شده است [۲]:

$$\langle l, \dots ; \xi, \eta, - \rangle \prec \langle l, \dots ; \xi', \eta', - \rangle$$
$$iff\ \xi < \xi' \ \vee (\xi = \xi' \wedge \eta < \eta')$$

۳-۲-۲-٦- مدل توأم برچسب‌زنی اجزای سخن و تجزیه

در این بخش، مدل‌هایی را که به صورت توأم، برچسب‌زنی اجزای سخن و تجزیه‌ی وابستگی را حل می‌کنند توضیح داده می‌شود که مبتنی بر تجزیه‌گرهای جابه‌جایی-کاهشی است که توضیح داده شد. در رابطه با دو تجزیه‌گر Parser-HS و $\text{Parser} - \text{ZN}^-$، دو مدل توأم بررسی می‌شود، $Joint - HS^+$ و $Joint - ZN^-$. گرچه دومی از مجموعه‌ی غنی‌تری از ویژگی‌ها استفاده می‌کند، البته قبلی‌ها از DP فایده‌ی بیشتری می‌برند زیرا ارائه‌ی فشرده‌ی ویژگی‌ها نتایج را در بسته‌بندی حالت با تکرار بیشتر انجام می‌دهد[۲].

۳-۲-۲-۷- برچسب‌زنی اجزای سخن با اقدام جابه‌جایی اصلاح شده[٤٦]

در مرجع [۲] آمده است: تجزیه‌گرهای توأم برچسب‌زنی اجزای سخن را در طول دوره‌ی تجزیه‌ی جابه‌جایی-کاهش با اصلاح اقدام شیفت ترکیب می‌کنند بنابراین یک برچسب را زمانی که کلمه جابه‌جا شد به آن نسبت می‌دهد.

- SHIFT(t) (SH(t)): سر صف، q_0، را به پشته جابه‌جا می‌کند و برچسب t را به آن نسبت می‌دهد.

همراه با اقدام‌های REDUCE-LEFT/RIGHT این مدل توأم از تعدادی از برچسب‌ها از مجموعه‌ی داده‌ی داده شده استفاده می‌کند.

[46] Modified Shift

۳-۲-۲-۸- آموزش و رمزگشایی

وظیفه‌ی برچسب‌زنی اجزای سخن و تجزیه‌ی وابستگی در یک چارچوب توأم فرموله می‌شود: در جمله‌ی داده شده‌ی ورودی تقسیم‌بندی شده‌ی x، مدل سعی می‌کند که بهترین خروجی yای را پیدا کند که عبارت زیر را ارضا کند.

$$\tilde{y} = arg \max_{y \epsilon Y(x)} \vec{w}\, \vec{\theta}(y),$$

که $Y(x)$ مجموعه‌ی خروجی‌های ممکن برای x است، $\vec{w}$ بردار ویژگی سراسری است، و $\vec{\theta}(y)$ بردار ویژگی y است. مانند تجزیه‌گرهای baseline، مدل‌ها با پرسپترون میانگین آموزش داده می‌شود؛ جستجوی پرتو و استراتژی به‌روز رسانی اولیه تقریبا یکسان‌اند به جز اینکه به‌روز رسانی با خطا در برچسب‌زنی اجزای سخن و همچنین خطا در تجزیه‌ی وابستگی ایجاد شده است. به‌طور مشابه برچسب‌زن baseline، از واژه‌نامه‌ی برچسب و برچسب‌های مجموعه‌ی بسته استفاده می‌کند تا برچسب‌های غیر محتمل را در طول به‌روز رسانی حذف کند[۲].

۳-۲-۲-۹- ویژگی‌ها

برای ویژگی‌های این مدل‌ها، اجتماعی از ویژگی‌ها در برچسب‌زن baseline و تجزیه‌گرهای baseline ترکیب می‌شود؛ ویژگی‌های Parser-HS برای Joint-HS و ویژگی‌هایی از $Parser - ZN^{-}$ برای $Joint - ZN^{-}$ مورد استفاده قرار گرفته‌اند. علاوه بر این، مجموعه‌ای از ویژگی‌های نحوی برای برچسب‌زنی اجزای سخن نیز ترکیب شده است که وابستگی‌های بین اجزای نحوی را در پشته و اجزای سخن‌هایی که برچسب می‌خورند را در نظر می‌گیرد.

در چارچوب توأم، وقتی مدل سعی می‌کند تا اقدام بعدی را تشخیص دهد برچسب‌های اجزای سخن پیش‌بینی شده در دسترس نیستند. به منظور مقابله با این موضوع، مفهوم ویژگی‌های تأخیری معرفی می‌شود که مدل را قادر می‌سازد تا اطلاعات پیش‌بینی را با ارزیابی تأخیری امتیازهای ویژگی‌ها ترکیب کند. توجه داشته باشید که ویژگی‌های تجزیه‌گرهای baseline برای همه‌ی اقدام‌ها (یعنی SHIFT(t) و REDUCE-LEFT/RIGHT) استفاده می‌شود در حالی که ویژگی‌های برچسب‌زن تنها برای

اقدام‌های REDUCE-LEFT/RIGHT در مدل‌های توأم استفاده شده است. و ویژگی‌های برچسب‌زنی به تعداد کمی عناصر جدید احتیاج دارد که تا به مجموعه‌ی ویژگی‌های کرنل اضافه شود؛ مجموعه‌ی جدید ویژگی‌های کرنل برای $Joint - HS^{+}$ در جدول ۳(c) نشان داده شده است [۲].

(a)	$q_0.t$	$q_0.w \circ q_0.t$	$s_0.t \circ q_0.t$	
	$s_0.t \circ q_0.t \circ q_1.t$		$s_1.t \circ s_0.t \circ q_0.t$	
	$s_0.w \circ q_0.t \circ q_1.t$		$s_1.t \circ s_0.w \circ q_0.t$	
(b)	$t \circ s_0.w$		$t \circ s_0.t$	
	$t \circ s_0.w \circ q_0.w$		$t \circ s_0.t \circ q_0.w$	
	$t \circ B(s_0.w) \circ q_0.w$		$t \circ E(s_0.w) \circ q_0.w$	
	$t \circ s_0.t \circ s_0.\text{rc}.t$		$t \circ s_0.t \circ s_0.\text{lc}.t$	
	$t \circ s_0.w \circ s_0.t \circ s_0.\text{rc}.t$		$t \circ s_0.w \circ s_0.t \circ s_0.\text{lc}.t$	
(c)	j	$s_2.t$ $q_0.w$	$q_{-1}.t$	$q_{-2}.t$
	$s_1.w$	$s_1.t$	$s_1.\text{rc}.w$	$s_1.\text{lc}.t$
	$s_0.w$	$s_0.t$	$s_0.\text{rc}.w$	$s_0.\text{lc}.t$

جدول ۳-(a) لیست ویژگی‌های تأخیری برای تجزیه‌گرهای توأم.(b) ویژگی‌های نحوی، برای تجزیه‌گرهای توأم، که t برچسب اجزای سخن ای است که به q_0 نسبت داده شده است. (c) ویژگی‌های کرنل برای تجزیه‌گر توأم $Joint - HS^{+}$ [۲].

به صورت ویژه، $q_{-1}.t$ و $q_{-2}.t$ به منظور تطبیق بعضی از ویژگی‌های برچسب‌زنی اضافه می‌شود، در حالی که $q_0.t$ و $q_1.t$ به دلیل برچسب‌های اجزای سخن پیش‌بینی شده زمانی که معادل بودن حالت‌ها ارزیابی می‌شود، در دسترس نیستند. $Joint - ZN^{-}$ علاوه بر این به ویژگی‌های کرنل در جدول ۲(d) نیاز دارد [۲].

۳-۲-۲-۹-۱-ویژگی‌های تأخیری

یک چالش در رویکرد توأم افزایشی این است که چون مدل جابه‌جایی-کاهش یک جمله‌ی ورودی را به روش چپ به راست پردازش می‌کند، نمی‌تواند از برچسب‌های اجزای سخن پیش‌بینی شده بهره‌برداری کند که یک تجزیه‌گر جابه‌جایی-کاهش پایپلاین می‌تواند آن‌ها را برای مشخص کردن گام بعدی در نظر بگیرد. به منظور از بین بردن این مشکل، مفهوم ویژگی‌های تأخیری معرفی می‌شود که مجموعه‌ای از ویژگی‌هاست که بعدا زمانی که اطلاعات خاص در دسترس قرار بگیرد ارزیابی می‌شود.

در مدلی که اینجا بحث می‌شود، ویژگی‌های تجزیه‌گر به اطلاعات اجزای سخن پیش‌بینی که به عنوان ویژگی‌های تأخیری تعریف شده است نیاز دارد، و بعدا در گامی که اجزای سخن پیش‌بینی شده مشخص می‌شود ارزیابی می‌گردد [۲].

در مثالی که شکل ۱۰ در [۲] آمده است، در گام ۲، یک تجزیه‌گر با یک کشمکش جابه‌جایی-کاهش مواجه می‌شود: اقدام بعدی می‌تواند هر کدام از REDUCE-LEFT/RIGHT و SHIFT(t) باشد. اگر این یک تجزیه‌گر جابه‌جایی-کاهش (غیر توأم) بود، مدل می‌تواند از اطلاعات اجزای سخن پیش‌بینی با ویژگی‌هایی نظیر زیر برای مشخص کردن اقدام بعدی استفاده می‌کند:

$$(s_0.t = VV)\ o\ (s_1.t = PN)\ o\ (q_0.t = BA),$$

زیرا اجزای سخن‌های همه‌ی کلمات در جمله، قبلا داده شده است. گرچه، در تجزیه‌گرهای توأم، اجزای سخن اولین کلمه در صف، 把، تا زمانی که کلمه جابه‌جایی پیدا کند نامشخص باقی می‌ماند. برای مقابله با این، ویژگی‌های تأخیری تعریف می‌شود که برچسب‌های اجزای سخن پیش‌بینی شده به عنوان آرگومان مانند زیر تعریف می‌شود.

$$(s_0.t = VV)\ o\ (s_1.t = PN)\ o\ (q_0.t = \lambda_1), \lambda_1 = w_2.t.$$

در گام ۳، بعد از اینکه SHIFT(BA) انجام شود، ویژگی‌های تأخیری گام قبلی یک ویژگی غیر تأخیری می‌شود.

$$(s_0.t = VV)\ o\ (s_1.t = PN)\ o\ (q_0.t = BA),$$

که می‌تواند به روش مشابهی مانند ویژگی‌های معمول(غیر تأخیری) ارزیابی شود.

step	action	stack S	queue Q	translation
0	-	ϕ	我/? 想/? 把/? ...	
1	SH(PN)	我/PN	想/? 把/? 这/? ...	我: "I"
2	SH(VV)	我/PN 想/VV	把/? 这/? 个/? ...	想: "want"
3	SH(BA)	我/PN 想/VV 把/BA	这/? 个/? 句子/? ...	把: *object marker*
4	SH(DT)	我/PN 想/VV 把/BA 这/DT	个/? 句子/? 翻译/? ...	这: "this"
5	SH(M)	我/PN 想/VV 把/BA 这/DT 个/M	句子/? 翻译/? 成/? ...	个: *quantifier*
6	RL	我/PN 想/VV 把/BA 这/DT⌒[个/M]	句子/? 翻译/? 成/? ...	
7	SH(NN)	我/PN 想/VV 把/BA 这/DT⌒[个/M] 句子/NN	翻译/? 成/? 英语/?	句子: "sentence"
8	RR	我/PN 想/VV 把/BA [这/DT⌒[...]]⌒句子/NN	翻译/? 成/? 英语/?	
9	RL	我/PN 想/VV 把/BA⌒[[...]⌒句子/NN]	翻译/? 成/? 英语/?	
10	SH(VV)	我/PN 想/VV 把/BA⌒[[...]⌒句子/NN] 翻译/VV	成/? 英语/?	翻译: "translate"

شکل ۱۰- اثر تجزیه‌ی توأم جابه‌جایی-کاهش [۲]

به صورت رسمی‌تر هر حالت، مجموعه‌ای از بردارهای ویژگی تأخیری $\langle \vec{d}_1, \vec{d}_2 \rangle$ را با خود حمل می‌کند، که n آرگومان بایستی پر شود[۴۷]. در هر گام، یک اقدام REDUCE-LEFT/RIGHT a مجموعه‌ی ویژگی‌های تأخیری را به بردارهای ویژگی تأخیری حالت ψ اضافه می‌کند:

$$\langle \vec{d}_1, \vec{d}_2 \rangle \leftarrow \langle \vec{d}_1 + \vec{\Phi}_1(\psi, a), \vec{d}_2 + \vec{\Phi}_2(\psi, a) \rangle,$$

که $\vec{\Phi}_1(\psi, a)$ و $\vec{\Phi}_2(\psi, a)$ ویژگی‌های تأخیری مرحله‌ی اول یا دومی هستند که تولید شده‌اند و با اقدام a به ψ اعمال می‌شوند. زمانی که اقدام SHIFT(t) اجرا می‌شود، مدل با ویژگی‌های تأخیری با برچسب t که به تازگی نسبت داده شده است پر می‌شود، همچنین ویژگی‌های تأخیری جدید را که تولید می‌کند اضافه می‌کند:

$$\langle \vec{d}_1, \vec{d}_2 \rangle \leftarrow \langle \vec{\Phi}_1\left(\psi, \mathrm{SH(t)}\right) + \tau(t, \vec{d}_2), \vec{\Phi}_2\left(\psi, \mathrm{SH(t)}\right) \rangle,$$

که $\tau(t, \vec{d}_2)$ بردار ویژگی به دست آمده بعد از این است که برچسب t با آرگومان اول ویژگی‌ها در $\vec{d}_2$ پر شده است. توجه داشته باشید که اقدام SH(t) همچنین $\vec{d}_0 = \tau(t, \vec{d}_1)$ را به بردار ویژگی‌اش (غیر تأخیری) وصل می‌کند.

[۴۷] در این مدل توأم ، تنها استفاده از بردارهای تأخیری مرتبه‌ی ۱ و مرتبه‌ی ۲ کافی است، زیرا الگوهای ویژگی به برچسب‌های دو کلمه‌ی اول اشاره می‌کند.

توجه داشته باشید که اگر رمزگشایی دقیق اجرا شود فرمول فوق با ویژگی‌های تأخیری، معادل مدلی با ویژگی‌های پیش‌بینی کامل است.

۳-۲-۲-۱۰- سیستم کاهش با برنامه‌نویسی پویا

با ویژگی‌های تأخیری، یک حالت تجزیه‌گر ψ شکل $\langle l, i, j, S, \vec{d}_1, \vec{d}_2; \xi, \eta, \Pi \rangle$ را به خود می‌گیرد. هم اکنون، اگر دو حالت مساوی بر اساس معادله‌ی ۱ با هم ترکیب شوند، یک حالت ممکن است چندین مجموعه‌ی بردارهای ویژگی تأخیری داشته باشد که به دنباله‌های اقدام قبلی بستگی دارد. به منظور ایجاد مدل توأمی که هنوز با شکل DP سازگار باشد، شرایط هم‌ارزی در معادله‌ی ۱ اصلاح می‌شود: علاوه بر شرط معادله‌ی ۱، دو حالت احتیاج دارند تا بردارهای ویژگی تأخیری یکسانی را به منظور ادغام آن‌ها به اشتراک بگذارند.

$$j = j' \wedge \vec{f}(j, S) = \vec{f}(j', S') \wedge \vec{d}_1 = \overrightarrow{d'}_1 \wedge \vec{d}_2 = \overrightarrow{d'}_2.$$

این تضمین می‌کند که حالت تجزیه‌گر تنها یک مجموعه‌ی یکتا از بردارهای ویژگی تأخیری را دارد. با کاهش می‌شود ثابت کرد که صحت حتی با ترکیب ویژگی‌های تأخیری تضمین شده است (اثبات با توجه به فضای محدود حذف شده است). گرچه، چون هر تعداد REDUCE-LEFT/RIGHT می‌تواند بین دو اقدام SHIFT(t) صورت بگیرد، ممکن است ویژگی‌های تأخیری نیاز داشته باشند تا به عناصر عمیق بیکران در درخت‌های پشته اشاره کنند. بنابراین، کران‌دار بودن ویژگی‌های کرنل دیگر نگه داشته نمی‌شود و در بدترین حالت، پیچیدگی چندجمله‌ای تضمین نمی‌شود[۲].

۳-۳- مدل‌های توأم مبتنی بر گراف

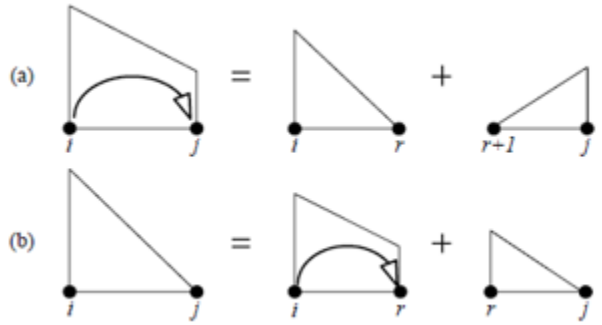
در این بخش چندین مدل توأم مبتنی بر گراف را مورد بررسی قرار می‌دهیم. یکی از چالش‌هایی که در مدل‌های مبتنی بر گراف وجود دارد بزرگ بودن گراف است. در همین راستا الگوریتم‌های هرس مختلفی هم برای هرس برچسب‌های اجزای سخن و هم برای هرس وابستگی‌های نامحتمل در هر کدام از روش‌ها ارائه شده است که برچسب‌های اجزای سخن و وابستگی‌های نامحتمل و یا با احتمال کمتر را هرس می‌کند. یکی دیگر از مشکلات این است که در مدل‌های مبتنی بر گراف عموما دقت برچسب‌زنی به نسبت دقت تجزیه‌ی وابستگی خیلی بالا نمی‌رود، زیرا ویژگی‌های نحوی بر ویژگی‌های اجزای سخن

مسلط‌اند. تلاش‌هایی برای حل این مشکل شده است که در ادامه خواهید دید. در همین راستا سعی کرده‌اند وزن الگوریتم‌های اجزای سخن را بالا ببرند تا تأثیر آن دیده شود. یکی دیگر از مسائل مهم برای مدل‌های توأم مبتنی بر گراف الگوریتم‌های رمزگشایی است تا ویژگی‌های غنی را در نظر بگیرد و از یک فضای فرضیه‌ی بزرگ نتایج بهینه را بدست بیاورد.

مشابه مدل‌های تجزیه‌ی وابستگی پایه که در فصل قبل شرح داده شد، مدل‌های توأم مرتبه اول، مرتبه دوم و مرتبه سوم براساس ویژگی‌های نحوی که در $f_{syn}(.)$ وجود دارد تعریف می‌شود.

۳-۳-۱- مدل‌های توأم مبتنی بر گراف نسخه‌ی ۱

مسئله‌ی ضروری برای روش توأم، طراحی الگوریتم‌های رمزگشایی کارا است تا ویژگی‌های غنی را در نظر بگیرد و نتایج بهینه از یک فضای فرضیه‌ی بزرگ را جستجو کند. [۲۰] یک ایده‌ی مقدماتی را برای اداره‌ی چندمعنایی[48] با گسترش الگوریتم‌های تجزیه توضیح می‌دهد. بر اساس این ایده، در اینجا الگوریتم‌های رمزگشایی [۱٤] و [۱۷] گسترش داده می‌شود و دو الگوریتم رمزگشایی مبتنی بر برنامه‌نویسی پویا (DP) برای مدل‌های توأم نسخه‌ی ۱ پیشنهاد می‌شود[1] .



شکل ۱۱- ساختارهای DP و مشتقات مرتبه اول الگوریتم رمزگشایی مدل‌های توأم نسخه‌ی ۱ [۱].

نسخه‌های دارای سر راست برای اختصار حذف شده‌اند. ذوزنقه spanهای ناکامل و مثلث spanهای کامل را نشان می‌دهد. دایره‌های توپر برچسب‌های اجزای سخن اندیس‌های نظیر را نشان می‌دهد.

۳-۳-۱-۱-الگوریتم رمزگشایی ۰۱:

همانطور که در شکل ۱۱ نشان داده شد، الگوریتم رمزگشایی توأم مرتبه‌ی اول از دو نوع از ساختارهای برنامه‌نویسی پویا استفاده می‌کند.

[48] Polysemy

(۱) Spanهای ناکامل شامل وابستگی و منطقه‌ی بین سر و فرزند

(۲) spanهای کامل شامل کلمه‌ی سر و فرزندانش در یک طرف

هر span به ترتیب با ترکیب دو span کوچک‌تر و همسایه در مد پایین به بالا ایجاد می‌شود.

شبه کدها در الگوریتم ۱ نشان داده شده است. $I_{(i,j)(t_i,t_j)}$ یک span غیر کامل از i به j را مشخص می‌کند که محدوده‌ی برچسب‌های اجزای سخن ، t_i و t_j است. $C_{(i,j)(t_i,t_j)}$ به یک span کامل از i به j اشاره می‌کند که محدوده‌ی برچسب‌های اجزای سخن آن t_i و t_j است. در مقابل، $I_{(j,i)(t_j,t_i)}$ و $C_{(j,i)(t_j,t_i)}$، spanهای جهت دیگر را ارائه می‌کند. توجه کنید که در این نگارش‌ها اندیس آرگومان اول همیشه به سر span اشاره می‌کند.

خط ٦ به مشتق در شکل ۱۱(a) اشاره می‌کند.

$Score_{joint}(x, t_i, t_r, t_{r+1}, t_j, p = \{(i,j)\})$ ویژگی‌های توأم ایجاد شده با ترکیب را در نظر می‌گیرد که $p = \{(i,j)\}$ بدین معنی است که بخش امتیازدهی که جدیدا مشاهده شده است وابستگی (i,j) است. ویژگی‌های نحوی که با $f_{syn}(x, t_i, t_j, i, j)$ مشخص می‌شود می‌تواند ویژگی‌های یونیگرام و بایگرام نحوی را ترکیب کند. ویژگی‌های اطراف و بین غیر قابل دسترس‌اند، زیرا برچسب‌های اجزای سخن متنی نظیر t_b و t_{i-1} در ساختارهای DP شامل نشده است.

1: $\forall 0 \le i \le n, t_i \in \mathcal{T}$ $C_{(i,i)(t_i,t_i)} = 0$ ◁ initialization
2: **for** $w = 1..n$ **do** ◁ span width
3: **for** $i = 0..(n-w)$ **do** ◁ span start index
4: $j = i + w$ ◁ span end index
5: **for** $(t_i, t_j) \in \mathcal{T}^2$ **do**
6: $I_{(i,j)(t_i,t_j)} = \max_{i \le r < j} \max_{(t_r,t_{r+1}) \in \mathcal{T}^2} \{C_{(i,r)(t_i,t_r)} + C_{(j,r+1)(t_j,t_{r+1})} + Score_{\text{joint}}(\mathbf{x}, t_i, t_r, t_{r+1}, t_j, p = \{(i,j)\})\}$
7: $I_{(j,i)(t_j,t_i)} = \max_{i \le r < j} \max_{(t_r,t_{r+1}) \in \mathcal{T}^2} \{C_{(i,r)(t_i,t_r)} + C_{(j,r+1)(t_j,t_{r+1})} + Score_{\text{joint}}(\mathbf{x}, t_i, t_r, t_{r+1}, t_j, p = \{(j,i)\})\}$
8: $C_{(i,j)(t_i,t_j)} = \max_{i < r \le j} \max_{t_r \in \mathcal{T}} \{I_{(i,r)(t_i,t_r)} + C_{(r,j)(t_r,t_j)} + Score_{\text{joint}}(\mathbf{x}, t_i, t_r, t_j, p = \emptyset)\}$
9: $C_{(j,i)(t_j,t_i)} = \max_{i \le r < j} \max_{t_r \in \mathcal{T}} \{C_{(r,i)(t_r,t_i)} + I_{(j,r)(t_j,t_r)} + Score_{\text{joint}}(\mathbf{x}, t_i, t_r, t_j, p = \emptyset)\}$
10: **end for**
11: **end for**
12: **end for**

الگوریتم ۱–الگوریتم رمزگشایی مرتبه اول نسخه‌ی ۱

بنابراین، ویژگی‌های شبه اطراف و بین به سادگی با ثابت کردن برچسب‌های اجزای سخن متنی به عنوان آن‌هایی که بیشترین احتمال را دارند، در نظر گرفته می‌شوند. با در نظر گرفتن ویژگی‌های بین به عنوان یک مثال، $t_i \hat{t}_b t_j dir\ dist$ مورد استفاده قرار می‌گیرد که $\hat{t}_b$ برچسب ۱–بهترینی است که با برچسب‌زن اجزای سخن پایه[۴۹] مشخص شده است. ویژگی‌های اجزای سخن، که با

[49] baseline

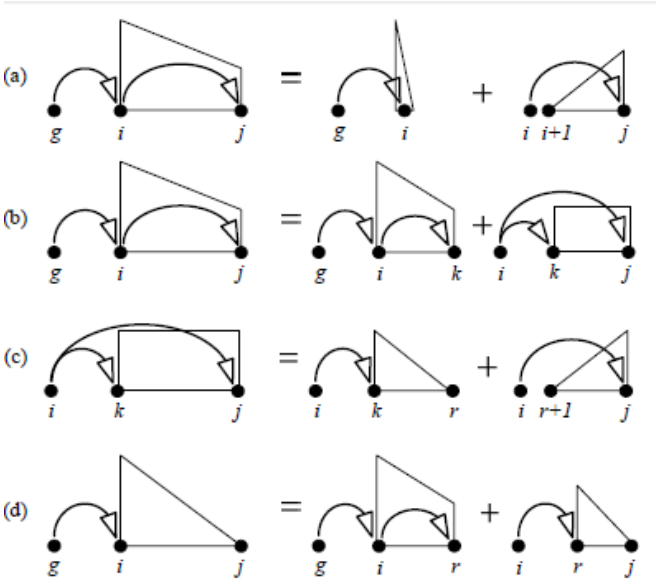
$f_{pos}(x, t_i, t_r, t_{r+1}, t_j)$ مشخص می‌شود تنها می‌تواند ویژگی‌های یونیگرام و بایگرام را ترکیب کند[۵۰]. به طور مشابه از شبه ویژگی‌های اجزای سخن تریگرام نظیر $\hat{t}_{r-1}t_r t_{r+1}$ استفاده می‌شود. خط ۸ مربوط به مشتق در شکل ۱۱(b) است. چون ساخته‌های ترکیبی هیچ بخش امتیازدهی‌ای ندارند(P=∅)، $Score_{joint}(x, t_i, t_r, t_j, p = \emptyset)$ تنها از ویژگی‌های اجزای سخن تشکیل شده است[۵۱].
خط ۷ و خط ۹ spanهایی را در جهت‌های مخالف ایجاد می‌کند. فضا و پیچیدگی الگوریتم $O(n^2q^2)$ و $O(n^3q^4)$ است که $q = |\tau|^5$.

۳-۳-۱-۲- الگوریتم رمزگشایی ۰۲ و ۰۳:

شکل ۱۲ الگوریتم رمزگشایی مرتبه‌ی ۲ و مرتبه‌ی ۳ مدل‌های توأم نسخه‌ی ۱ را نشان می‌دهد. یک نوع جدید از span که sibling span نامیده می‌شود ساختارهای sibling را مد نظر قرار می‌دهد. به علاوه به هر span یک اندیس پدربزرگ افزوده می‌شود تا ساختارهای grandparent و grand-sibling را بگیرد. شبه ویژگی‌های اطراف، بین و تریگرام اجزای سخن به دلیل بالا مورد استفاده قرار می‌گیرد. پیچیدگی فضایی و زمانی الگوریتم‌ها به ترتیب $O(n^3q^3)$ و $O(n^4q^5)$ است.

۵۰- ۱- $w_r t_r$ اگر $i \neq r$ ۲- $w_{r+1}t_{r+1}$ اگر r+1≠ j ۳- $t_r t_{r+1}$ اگر $r \neq i$ یا r+1≠ j ۴- $t_i t_r$ اگر r-1=i ۵- $t_{r+1}t_j$ اگر r+2=j. توجه داشته باشید که $w_i t_i$ و $w_j t_j$ و $t_i t_j$(اگر i=j-1) برای پرهیز از شمارش مجدد ترکیب نمی‌شوند.

۵۱- ۱- $w_r t_r$ اگر $r \neq j$، ۲- $t_i t_r$ اگر i=r-1، ۳- $t_r t_j$ اگر r+1=j به همین شکل شبه ویژگی‌های تریگرام نیز می‌تواند افزوده شود.



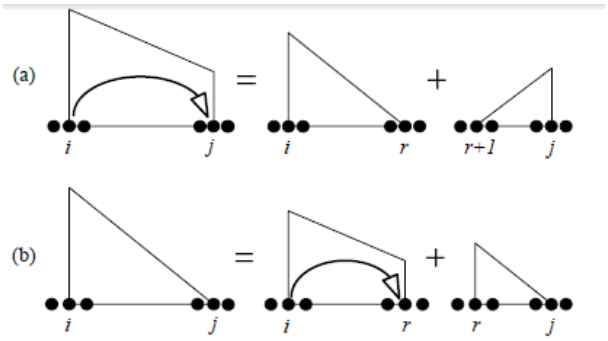
شکل ۱۲- ساختارهای DP و مشتقات مرتبه‌ی دوم و سوم الگوریتم رمزگشایی توأم نسخه‌ی ۱. برای اختصار، نسخه‌ی سر راست و پدربزرگ راست حذف شده است. مستطیل sibling span را نشان می‌دهد.

۳-۳-۲-مدل‌های توأم نسخه‌ی ۲

برای ترکیب بیشتر ویژگی‌های نحوی اطراف واقعی و تریگرام اجزای سخن در ساختارهای DP الگوریتم مدل‌های توأم نسخه‌ی ۱ گسترش می‌یابد و مدل توأم نسخه‌ی ۲ پیشنهاد می‌شود [۱].

۳-۳-۲-۱- الگوریتم رمزگشایی ۰۱:

شکل ۱۳ الگوریتم رمزگشایی توأم مرتبه‌ی اول نسخه‌ی ۲ را نشان می‌دهد. در مقایسه با ساختارهای موجود در شکل ۱۱هر span با برچسب‌های اجزای سخن اطراف اندیس‌های مرزی اضافه می‌شود. این برچسب‌های اجزای سخن متنی $Score_{joint}(.)$ در خط‌های ٦-۹ الگوریتم ۱ می‌توانند ویژگی‌های نحوی اطراف و تریگرام اجزای سخن را بگیرند، اما به شمارش برچسب‌های اجزای سخن بر روی تعداد زیادی اندیس احتیاج است. برای اختصار از شبه کدهایی که می‌توانند از الگوریتم ۱ به دست بیایند صرف نظر شده است. پیچیدگی فضایی و زمانی الگوریتم به ترتیب $O(n^2q^6)$ و $O(n^3q^{10})$ است.



شکل ۱۳- ساختارهای DP و مشتقات مرتبه‌ی اول الگوریتم رمزگشایی نسخه‌ی ۲. برای سادگی نسخه‌ی سر راست حذف شده است.

۳-۳-۲-۲- الگوریتم رمزگشایی ۰۲ و ۰۳:

با استفاده از ایده‌ای مشابه بالا، الگوریتم رمزگشایی توأم مرتبه ۲ و ۳ در نسخه‌ی ۲ می‌تواند بر اساس شکل ۱۲ گرفته شود. پیچیدگی فضایی و زمانی به ترتیب $O(n^3q^7)$ و $O(n^4q^{11})$ است.
ویژگی‌های بین که در موقعیت توأم، به عنوان ویژگی‌های غیر محلی در نظر گرفته می‌شوند هنوز نمی‌توانند با مدل‌های توأم نسخه‌ی ۲ ترکیب شوند[۱].

۳-۳-۲-۳- مقایسه

بر اساس مثال بالا، مدل‌های توأم نسخه‌ی ۱ با در نظر گرفتن تعداد برچسب‌های اجزای سخن برای هر کلمه بسیار کاراترند اما موفق به ترکیب ویژگی‌های نحوی اطراف و ویژگی‌های تریگرام اجزای سخن در ساختارهای DP نشده‌اند. مدل‌های توأم نسخه‌ی ۲ می‌توانند مجموعه‌های ویژگی فوق را ترکیب کنند، اما پیچیدگی بالایی دارند.

۳-۳-۲-٤-هرس برچسب اجزای سخن

در [۱] برای هرس برچسب اجزای سخن این روش پیشنهاد شده است: پیچیدگی زمانی الگوریتم رمزگشایی توأم با در نظر گرفتن تعداد برچسب‌های اجزای سخن نامزد برای هر کلمه به نحو تحمل ناپذیری بالاست($q=|\tau|$). زمانی که تنها از دو تا از محتمل‌ترین برچسب‌های اجزای سخن برای هر کلمه ($q=2$) استفاده می‌شود حتی برای مدل‌های توأم نسخه‌ی ۱ زمان بسیار زیادی مورد استفاده قرار می‌گیرد. برای حل این مسئله، یک روش هرس که به صورت موثری فضای برچسب اجزای سخن را براساس مدل برچسب‌زنی احتمالی کاهش می‌دهد پیشنهاد می‌گردد.

یک مدل خطی لگاریتمی در نظر گرفته می‌شود که یک توزیع شرطی دنباله‌ی برچسب اجزای سخن t با x داده شده به صورت زیر است:

$$P(t|x) = \frac{e^{w_{pos} \cdot f_{pos}(x,t)}}{\sum_t e^{w_{pos} \cdot f_{pos}(x,t)}}$$

از مجموعه‌ی ویژگی‌های مشابه f_{pos} تعریف شده در بخش ۲-۱-۲ و از الگوریتم گرادیان نمایی برای یادگیری بردار وزن w_{pos} استفاده می‌شود. احتمال حاشیه‌ای برچسب‌زنی یک کلمه‌ی w_i به عنوان t به صورت زیر است:

$$P(t_i = t|x) = \sum_{t:t[i]\equiv t} P(t|x)$$

که می‌تواند به صورت کارایی با استفاده از الگوریتم رو به جلو-رو به عقب[۵۲] محاسبه شود. $\text{pmax}_i(x)$ به عنوان بیشترین احتمال حاشیه‌ای برچسب‌زنی کلمه‌ی w_{pos} تعریف می‌شود:

$$Pmax_i(x) = \max_{t\epsilon\tau} P(t_i = t|x)$$

سپس برچسب‌های اجزای سخن نامزد مجاز کلمه‌ی w_i به صورت زیر تعریف می‌شود:

$$\tau_i(x) = \{t: t\epsilon\, \tau\, , P(t_i = t|x) \geq \lambda_t \times Pmax_i(x)\}$$

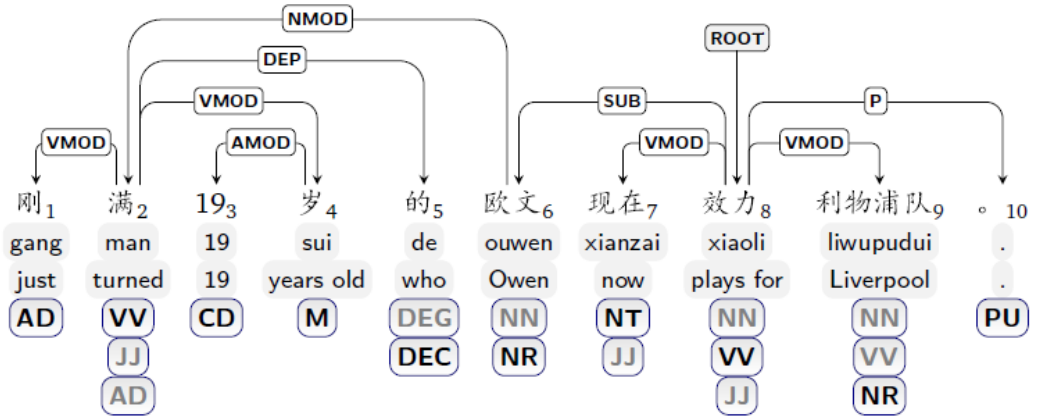
که λ_t آستانه‌ی هرس است. $\tau_i(x)$ برای محدود کردن فضای جستجوی اجزای سخن با جایگزینی τ در الگوریتم ۱ است.

۳-۳-۳- مدل توأم مبتنی بر گراف منفعل-پرخاشگر جداگانه[۵۳]

جمله‌ی ورودی با n کلمه داده شده است، که با $x = w_1 \dots w_2$ مشخص می‌شود. برچسب‌زن اجزای سخن(اجزای سخن) در نظر دارد تا دنباله‌ی برچسب بهینه $t = t_1 \dots t_n$ را پیدا کند که $(1 \leq i \leq t_i \epsilon\, \vartheta\, n)$ و ϑ مجموعه‌ای از برچسب‌های از پیش تعریف شده است. اندازه‌ی ϑ معمولا خیلی کمتر از اندازه‌ی واژگان است شکل ۱٤ جمله‌ی نمونه‌ای از پیکره‌ی درختی چینی پن را ارائه می‌کند. سه سطر پایین n-بهترین برچسب‌های اجزای سخن برای هر کلمه را ارائه می‌کند که با مدل CRF به‌روز تولید شده است. به برچسب‌های اجزای سخن اول بهترین نگاه کنید، می‌بینید که مدل CRF چهار خطا ایجاد می‌کند،یعنی، de/DEC→ DEG، ouwen/NR→ NN، xiaoli/VV→ NN و

[52] forward-backward
[53] Separately Passive-Aggressive

liwupudui/NR→ NN. در حقیقت، ابهامات (DEC,DEG) و (NN,VV)، که نیازمند به دانش زیادی برای حل هستند برای مدل‌های برچسب‌زنی ترتیبی بسیار مشکل‌اند.



شکل ١٤- مثالی برای CTB5. ترجمه‌های انگلیسی در دو سطر زیر جمله‌ی چینی ارائه شده است. برچسب‌های اجزای سخن نامحتمل را بر اساس احتمالات حاشیه‌ای(بخش ٤-١ را ببینید) هرس کرده و سه برچسب اجزای سخن بالای نامزد را در سه سطر لیست می‌کند(برچسب‌های اجزای سخن ناصحیح به رنگ خاکستری است و برچسب‌های صحیح به رنگ سیاه است)[٣].

تجزیه‌ی وابستگی یک جمله‌ی زبان طبیعی را به یک درخت وابستگی ساختاری منطبق با گرامر وابستگی از پیش تعریف شده همانطور که در شکل ١٤ نشان داده شد، نگاشت می‌کند. همانطور که قبلا گفته شد، یک درخت وابستگی با $d = \{(h,m,l): 1 \leq h \leq n, 1 \leq m \leq n, l \epsilon \gamma\}$ که، (h,m,l) یعنی یک وابستگی از کلمه‌ی سر(همچنین پدر نیز نامیده می‌شود) w_h به پیرایندە(همچنین فرزند یا وابسته نیز نامیده می‌شود) w_m با برچسب وابستگی l، و γ مجموعه‌ی برچسب است. برچسب‌های وابستگی برای نشان دادن روابط نحوی یا معنایی بین دو کلمه استفاده می‌شود. مدل‌های تجزیه‌ی وابستگی داده‌ای استفاده‌ی زیادی از برچسب‌های اجزای سخن برای ساختن ویژگی‌های پشتیبان می‌کنند، زیرا اگر ویژگی‌های لغوی به تنهایی استفاده شوند، این منجر به ایجاد مسئله‌ی جدی تنکی داده می‌شود. مدل‌های توأم می‌توانند به صورت قابل ملاحظه‌ای دقت تجزیه را تقویت کنند در مقابل زیروظیفه‌ی برچسب‌زنی سود زیادی از این چارچوب توأم نمی‌برد. این با این حقیقت که ساختار نحوی بهتر می‌تواند به ابهام‌زدایی اجزای سخن هم کمک کند در تناقض است. تجزیه و تحلیل خطا در [1] نشان می‌دهد که مدل‌های توأم

برای حل ابهامات اجزای سخن که حساس به نحواند نظیر {VV, NN} و {DEC, DEG} مهم است، اما در ابهام‌زدایی {NN, NR} و {NN, JJ} که نقش‌های مشابهی را در ساختارهای نحوی بازی می‌کنند بسیار ضعیف است. یک دلیل ممکن این است که مدل‌های توأم مبتنی بر گراف [۱] با ویژگی‌های نحوی تحت سلطه قرار گرفته است. به طور متوسط امتیاز متناظر با ویژگی‌های اجزای سخن تنها یک پنجاهم امتیاز ویژگی‌های نحوی در نتایج توأم بازگشتی است. به عبارت دیگر، ویژگی‌های اجزای سخن تأثیر کمی در تعیین بهترین نتیجه‌ی توأم دارد. بنابراین، مدل‌های توأم برچسب‌های اجزای سخن‌ای را پیشنهاد می‌دهد که از نقطه نظر تجزیه مفیدتر و متمایز است [۳].

۳-۳-۳-۱ الگوریتم منفعل-پرخاشگر جداگانه(SPA)

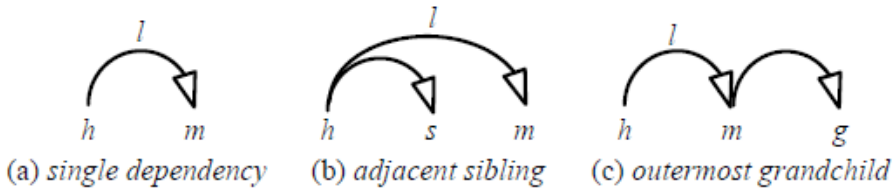
SPA به صورت جداگانه وزن‌های ویژگی اجزای سخن و وزن‌های ویژگی نحوی را به‌روز رسانی می‌کند و به صورت طبیعی وزن‌های ویژگی‌های اجزای سخن را با استفاده از چارچوب بهینه‌سازی توأم بالا می‌برد. به عنوان یک نتیجه، SPA می‌تواند باعث استفاده‌ی بهتری از قدرت تمایز ویژگی‌های اجزای سخن در حل ابهامات اجزای سخن حساس به نحو شود که این منجر به بهبود بسیار زیاد دقت برچسب‌زنی می‌شود. از سوی دیگر، بهبود دقت برچسب‌زنی می‌تواند باعث کمک بیشتر به تجزیه شود[۳].

روش پایپلاین با برچسب‌زنی اجزای سخن و تجزیه‌ی وابستگی به عنوان دو مشکل آبشاری رفتار می‌کند. اول، دنباله‌ی برچسب بهینه‌ی $\hat{t}$ مشخص می‌شود.

$$\hat{t} = arg \max_{t} Score_{pos}(x, t) \qquad (3)$$

سپس، درخت وابستگی بهینه $\hat{d}$ بر اساس x و $\hat{t}$ مشخص می‌شود.

$$\hat{\mathrm{d}} = \arg \max_{\mathrm{d}} \mathrm{Score}_{\mathrm{syn}}(\mathrm{x}, \hat{\mathrm{t}}, \mathrm{d}) \qquad (4)$$



شکل ۱۵- سه نوع از امتیازدهی زیردرخت استفاده شده در روش تجزیه‌ی و مدل‌های توأم [۳].

Feature category	Atomic features incorporated
Dependency features $\mathbf{f}_{\text{dep}}(\mathbf{x},\mathbf{t},h,m,l)$	$l, w_h, w_m, t_h, t_m, t_{h\pm1}, t_{m\pm1}, t_b, dir(h,m), dist(h,m)$
Sibling features $\mathbf{f}_{\text{sib}}(\mathbf{x},\mathbf{t},h,m,l,s)$	$l, w_h, w_s, w_m, t_h, t_m, t_s, t_{h\pm1}, t_{m\pm1}, t_{s\pm1}, dir(h,m), dist(h,m)$
Grandchild features $\mathbf{f}_{\text{grd}}(\mathbf{x},\mathbf{t},h,m,l,g)$	$l, w_h, w_m, w_g, t_h, t_m, t_g, t_{h\pm1}, t_{m\pm1}, t_{g\pm1}, dir(h,m), dir(m,g)$

جدول ٤- نمایش مختصری از ویژگی‌های نحوی. B اندیسی بین h و m است. dir(i,j) و dist(i,j) جهت و فاصله‌ی وابستگی i و j را نشان می‌دهد. لطفا برای مشاهده‌ی لیست ویژگی کامل به جدول ٤ مقاله‌ی [21] مراجعه کنید.

۳-۳-۳-۲- برچسب‌زنی اجزای سخن مبتنی بر CRF

CRF مرتبه‌ی اول برای ساختن برچسب‌زن اجزای سخن baseline در نظر گرفته می‌شود. به عنوان یک مدل احتمالی خطی-لگاریتمی شرطی، CRF احتمال یک دنباله‌ی برچسب را به صورت زیر تعریف می‌کند.

$$p\left(t|\mathrm{x}\right) = \exp\left(Score_{pos}(x,t)\right) / \sum_{t'} \exp(Score_{pos}(x,t'))$$

$$Score_{pos}(x,t) = w_{pos}.f_{pos}(x,t) = \sum_{1\le i\le n} w_{pu} f_{pu}(x,t_i) + w_{pb}.f_{pb}(x,t_{i-1},t_i) \qquad (5)$$

که $f_{pos,pu,pb}(.)$ به بردارهای ویژگی اشاره می‌کند و $w_{pos,pu,pb}$ متناظر بردارهای وزن است. $f_{pu}(\mathrm{x},t_i)$ بردارهای یونیگرام اجزای سخن است و $f_{pb}(\mathrm{x},t_{i-1},t_i)$ ویژگی‌های بایگرام اجزای سخن است. برای چینی، ویژگی‌هایی که توسط [۲۲] پیشنهاد شده است را در نظر گرفته می‌شود. آن‌ها از مشخصه‌های چینی که در یک کلمه نهفته است برای ساختن ویژگی‌های غنی استفاده می‌کنند، که

مشخص می‌شود برای کلمات با تکرار کم مناسب است. برای کلمات انگلیسی از ویژگی‌های [۷] بهره می‌برد که از پسوندها و پیشوندها برای بهبود کارایی برچسب‌زنی بر روی کلمات نادر استفاده می‌شود.

۳-۳-۳-۳-تجزیه‌ی گراف وابستگی مبتنی بر گراف مرتبه‌ی ۲

رویکرد مبتنی بر گراف تجزیه‌ی وابستگی را به صورت یافتن درخت با بیشترین امتیاز در یک گراف جهت‌دار در نظر می‌گیرد. در اینجا مدل مرتبه‌ی دوم [۱۶] ساخته شده است زیرا مطالعات گذشته نشان می‌دهد که منجر به بهترین دقت تجزیه بر روی زبان‌های گوناگون می‌شود[۲۱] [۱۷]. امتیاز درخت وابستگی در امتیازهای سه نوع زیردرخت به عنوان یک فاکتور در شکل ۱۵ نشان داده شده است.

$$\begin{aligned} Score_{syn}(x,t,d) &= w_{syn}.f_{syn}(x,t,d) \\ &= \sum_{\{(h,m,l)\}\subseteq \mathrm{d}} w_{dep}.f_{dep}(x,t,h,m,l) \\ &+ \sum_{\{(h,m,l),(h,s)\}\subseteq \mathrm{d}} w_{sib}.f_{sib}(x,t,h,m,l,s) \\ &+ \sum_{\{(h,m,l),(m,g)\}\subseteq \mathrm{d}} w_{grd}.f_{grd}(x,t,h,m,l,g) \end{aligned} \qquad (6)$$

برای ویژگی‌های نحوی، آن‌هایی که در [۲۱] بودند ساخته می‌شوند که شامل سه دسته‌ی متناظر با سه نوع از زیردرخت‌های امتیازدهی است. برای تجزیه‌ی بدون برچسب و مدل‌های توأم، برچسب l حذف می‌شود. در مقایسه با ویژگی‌های نحوی استفاده شده در [۱] این مجموعه‌ی ویژگی به بررسی برچسب‌های اجزای سخن متنی بیشتری شامل $t_{s\pm1}$ و $t_{g\pm1}$ می‌پردازد [۳].

۳-۳-۳-٤- رمزگشایی

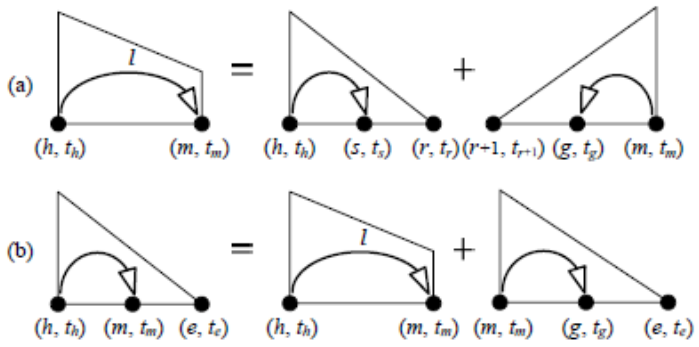
در این مدل توأم برای رمزگشایی الگوریتم تجزیه‌ی [۱۶] با ایده‌ی [۲۰] گسترش داده شده و یک برنامه‌نویسی پویا(DP) مبتنی بر الگوریتم رمزگشایی برای این مدل توأم پیشنهاد شده است. شکل ۱۶ ساختار DP پایه و عملیات آن را نشان می‌دهد. ایده‌ی کلیدی این است که به تقویت ساختارهای DP پایه در الگوریتم تجزیه (یعنی span) با تعداد کمی از برچسب‌های پایه پرداخته شود. span یعنی ساختاری که تا حدی یک زیر جمله را پوشش دهد. برای مثال، span سمت چپ در شکل ۱۶(a) که

span غیر کامل نامیده می‌شود و با $I_{(h,m,l)(t_h,t_m)}$ در نظر گرفته می‌شود، یک درخت جزئی پوشا $w_h \dots w_m$ را نشان می‌دهد که w_h با t_h برچسب خورده است و w_m با t_m. span سمت چپ در شکل ۱۶(b) یک span کامل است و با $C^{(e)(t_e)}_{(h,m)(t_h,t_m)}$ نشان داده می‌شود [۳].

الگوریتم رمزگشایی در مد پایین به بالا کار می‌کند و دو span کوچکتر را با یکی بزرگتر در هر مرحله ترکیب می‌کند. در طی ترکیب، ویژگی‌های تازه معرفی شده ترکیب می‌شوند و در نتیجه امتیاز span محاسبه می‌شود. برای مثال، عملیات در شکل ۱۶(a) ۵ مجموعه‌ی ویژگی را معرفی می‌کند یعنی،

$$f_{dep(x,t_h,t_m,h,m,l)}, f_{sib(x,t_h,t_m,t_s,h,m,l,s)}, f_{grd(x,t_h,t_m,t_g,h,m,l,g)}, f_{pu(x,t_m)}, f_{pb(x,t_r,t_{r+1})}$$

و عملیات در شکل ۱۶(b) یک مجموعه‌ی ویژگی $f_{grd(x,t_h,t_m,t_g,h,m,l,g)}$ را معرفی می‌کند. توجه داشته باشید که در توابع ویژگی نحوی بالا چندین برچسب اجزای سخن متن در ساختارهای DP رمز نشده‌اند و بنابراین در لیست‌های پارامتر ارائه نشده‌اند، که شامل $t_{h\pm1}, t_{m\pm1}, t_b, t_{s\pm1}, t_{g\pm1}$ می‌باشند. برای آن‌ها از برچسب‌های اجزای سخن با بیشترین احتمال که توسط مدل برچسب‌زنی CRF baseline تهیه شده است استفاده می‌شود. آن‌ها یافته‌اند که این تقریب به صورت قابل ملاحظه‌ای کارایی مدل‌های توأم را بدون از دست دادن دقت و صحت بهبود بخشد. پیچیدگی زمانی الگوریتم $O(|\gamma|n^4q^5)$ است و پیچیدگی فضایی از $O(|\gamma|n^2q^2 + n^3q^3)$ است که q شماره‌ی برچسب هر کلمه است($|\vartheta| \leq$)است [۳].



شکل ۱۶- یک مثال از برنامه‌نویسی پویا مبتنی بر الگوریتم رمزگشایی برای این مدل توأم است. برای اختصار ایجاد spanهای دست راست حذف شده است [۳].

۳-۳-۳-۵- الگوریتم آموزش منفعل-پرخاشگر جداگانه

الگوریتم ۲ که در [۳] ارائه شده است، چارچوب کلی آموزش آنلاین را نشان می‌دهد. آموزش آنلاین به صورت تکراری کل مجموعه‌ی داده‌ی آموزشی را پیمایش می‌کند و در هر دفعه از یک نمونه برای به‌روز رسانی وزن‌های ویژگی استفاده می‌کند. در ابتدا بهترین نتایج برای نمونه براساس وزن‌های ویژگی جاری (خط ٦) یافته می‌شود. سپس وزن‌های ویژگی با مقایسه‌ی بهترین نتایج و مرجع استاندارد طلایی به‌روز رسانی می‌شود(خط ۷). طبق شرط به‌روز رسانی، سه الگوریتم آموزشی متفاوت به طور گسترده‌ای در تجزیه مورد استفاده قرار می‌گیرند. یعنی، پرسپترون میانگین (AP) [۹]، الگوریتم منفعل-پرخاشگر (PA) [۲۳]، و MIRA[54] [۲٤].

1. **Input:** Training Data $\mathbb{D} = \{(\mathbf{x}^{(j)}, \mathbf{t}^{(j)}, \mathbf{d}^{(j)})\}_{j=1}^{N}$
2. **Output:** $\mathbf{w}_{\text{joint}}$ $(\equiv \mathbf{w}_{\text{pos}\oplus\text{syn}})$
3. **Initialization:** $\mathbf{w}_{\text{joint}}^{(0)} = 0; \mathbf{v} = 0; k = 0$
4. **for** $i = 1$ to I **do** // *iterations*
5. **for** $j = 1$ to N **do** // *traverse the samples*
6. $(\hat{\mathbf{t}}, \hat{\mathbf{d}}) = \arg\max_{\mathbf{t},\mathbf{d}} \mathbf{w}_{\text{joint}}^{(k)} \cdot \mathbf{f}_{\text{joint}}(\mathbf{x}^{(j)}, \mathbf{t}, \mathbf{d})$ // *decode based on current feature weights.*
7. $\mathbf{w}_{\text{joint}}^{(k+1)} = \text{update } \mathbf{w}_{\text{joint}}^{(k)} \text{ with } (\mathbf{x}^{(j)}, \mathbf{t}^{(j)}, \hat{\mathbf{t}}, \mathbf{d}^{(j)}, \hat{\mathbf{d}})$ // *update weights according to some criterion.*
8. $\mathbf{v} = \mathbf{v} + \mathbf{w}_{\text{joint}}^{(k+1)}$
9. $k = k + 1$
10. **end for**
11. **end for**
12. $\mathbf{w}_{\text{joint}} = \mathbf{v}/(I \times N)$ // *average the weights*

الگوریتم ۲- آموزش آنلاین کلی برای برچسب‌زنی اجزای سخن و تجزیه‌ی وابستگی توأم.

تمام الگوریتم‌ها جهت فاصله‌ی بین بردار ویژگی مرجع $f_{joint}\,(x^{(j)}, t^{(j)}, d^{(j)})$ و بردار ویژگی بهترین نتایج $f_{joint}(x^{(j)}, \hat{t}, \hat{d})$ را به‌روز رسانی می‌کنند. با این حال، استراتژی‌های مختلفی برای تشخیص گام به‌روز رسانی در نظر گرفته شده است. AP از گام به‌روز رسانی ثابت ۱ استفاده می‌کند.

$$AP\left\{w_{joint}^{(k+1)} = w_{joint}^{(k)} + f_{joint}(x^{(j)}, t^{(j)}, d^{(j)}) - f_{joint}(x^{(j)}, \hat{t}, \hat{d}) \right. \qquad (7)$$

[54] margin infused relaxed algorithm

PA گام به‌روز رسانی τ_{joint} را با در نظر گرفتن میزان loss بهترین نتایج، فاصله‌ی امتیاز، و فاصله‌ی بردار ویژگی محاسبه می‌کند.

$$PA \begin{cases} \tau_{joint} \\ w_{joint}^{(k+1)} \end{cases} \qquad (8)$$

$$\tau_{joint} = \frac{Score_{joint}(x^{(j)}, \hat{t}, \hat{d}) - Score_{joint}(x^{(j)}, t^{(j)}, d^{(j)}) + \rho_{pos}(t^{(j)}, \hat{t}) + \rho_{syn}(d^{(j)}, \hat{d})}{\left\| Score_{joint}(x^{(j)}, t^{(j)}, d^{(j)}) - Score_{joint}(x^{(j)}, \hat{t}, \hat{d}) \right\|^2}$$

$$w_{joint}^{(k+1)} = w_{joint}^{(k)} + \tau_{joint}\left(f_{joint}(x^{(j)}, t^{(j)}, d^{(j)}) - f_{joint}(x^{(j)}, \hat{t}, \hat{d})\right)$$

که $\rho_{pos}(t^{(j)}, \hat{t})$ تعداد برچسب اجزای سخن ناصحیح در $\hat{t}$ است و $\rho_{syn}(d^{(j)}, \hat{d})$ تعداد خطای وابستگی در $\hat{d}$ بر طبق $d^{(j)}$ است. $\rho_{syn}(d^{(j)}, \hat{d})$ برای یک وابستگی نادرست ۱ واحد افزایش پیدا می‌کند و برای هر وابستگی صحیح با برچسب غلط ۰,۵ واحد افزایش پیدا می‌کند. از لحاظ تئوری معادله‌ی ۸ کوچکترین به‌روز رسانی‌ای را محاسبه می‌کند که باعث می‌شود که فرضیه‌ی درست امتیاز بیشتری از فرضیه‌ای که بیشترین امتیاز را برمی‌گرداند و خطا کرده است داشته باشد.

AP و PA از گام به‌روز رسانی یکسان برای ویژگی‌های اجزای سخن $f_{pos}(.)$ و ویژگی‌های نحوی $f_{syn}(.)$ استفاده می‌کنند. بنابراین، وزن‌های ویژگی‌های اجزای سخن و ویژگی‌های نحوی بعد از کامل شدن آموزش دارای مقدارهای مساوی‌ای هستند. این مشکل‌ساز است زیرا تعداد ویژگی‌های نحوی بسیار بیشتر از تعداد ویژگی‌های اجزای سخن است. $f_{joint}(x^{(j)}, \hat{t}, \hat{d})$ و $f_{syn}(x^{(j)}, \hat{t}, \hat{d})$ بیش از ۳۰۰۰ ویژگی غیر صفر دارند، در حالی که $f_{pos}(x^{(j)}, \hat{t})$ به طور متوسط کمتر از ۲۰۰ ویژگی غیر صفر دارد. در نتیجه، مدل تحت سلطه‌ی ویژگی‌های نحوی قرار دارد و ویژگی‌های اجزای سخن نقش بسیار

محدودی در مشخص کردن بهترین نتیجه‌ی توأم دارد. در واقع، ویژگی‌های اجزای سخن زمانی که با AP و PA آموزش داده می‌شوند، کمک کوچکی به امتیازهای بهترین نتایج توأم می‌کنند.

الگوریتم آموزشی پیشنهاد شده الگوریتم جداگانه‌ی منفعل-پرخاشگر نامیده می‌شود. SPA دو گام به‌روزرسانی برای ویژگی‌های اجزای سخن و ویژگی‌های نحوی محاسبه می‌کند.

$$SPA\begin{cases}\tau_{pos}=\dfrac{Score_{pos}(x^{(j)},\hat{t})-Score_{pos}(x^{(j)},\,t^{(j)})+\rho_{pos}(\,t^{(j)},\hat{t})}{\left\|f_{pos}(x^{(j)},\,t^{(j)})-f_{pos}(x^{(j)},\hat{t})\right\|^2}\\ \tau_{syn}=\dfrac{Score_{syn}(x^{(j)},\hat{t},\hat{d})-Score_{syn}(x^{(j)},t^{(j)},d^{(j)})+\rho_{syn}(d^{(j)},\hat{d})}{\left\|f_{syn}(x^{(j)},\,t^{(j)},\,d^{(j)})-f_{syn}(x^{(j)},\hat{t},\hat{d})\right\|^2}\\ w_{pos}^{(k+1)}=w_{pos}^{(k)}+\tau_{pos}(f_{pos}(x^{(j)},\,t^{(j)})-f_{pos}(x^{(j)},\hat{t}))\\ w_{syn}^{(k+1)}=w_{syn}^{(k)}+\tau_{syn}(f_{syn}(x^{(j)},\,t^{(j)},d^{(j)})-f_{syn}(x^{(j)},\hat{t},\hat{d}))\end{cases}\qquad(9)$$

مشابه PA معادله‌ی ۹ به صورت جداگانه کوچکترین به‌روزآوری را برای وزن‌های ویژگی اجزای سخن پیدا می‌کند که امتیاز اجزای سخن فرضیه‌ی صحیح بیشتر از امتیاز اجزای سخن فرضیه‌ی با بیشترین امتیاز برگردانده شده با خطای اجزای سخن است، و کوچکترین به‌روزآوری وزن‌های ویژگی نحوی است که امتیاز نحوی فرضیه‌ی صحیح بیشتر از بیشترین امتیاز برگردانده شده‌ی فرضیه‌ای با خطای نحوی است. توجه کنید که معادله‌ی ۹ گاهی اوقات یک عدد منفی τ_{POS} یا τ_{syn} می‌شود، زیرا فرضیه‌ی صحیح از قبل دارای زیرامتیاز نحو یا اجزای سخن بیشتر از اول-بهترین است اما کارایی به صورت کامل به زیر امتیاز دیگری منحصر شده است. با این وجود، این موارد نادر هستند. وقتی این عدد منفی است گام به‌روز رسانی را صفر قرار داده می‌شود.

چون $f_{pos}(.)$ شامل ویژگی‌های غیر صفر کمتری از $f_{syn}(.)$ است. $\left\|f_{pos}(x^{(j)},\,t^{(j)})-f_{pos}(x^{(j)},\hat{t})\right\|^2$ خیلی کمتر از $\left\|f_{syn}(x^{(j)},\,t^{(j)},\,d^{(j)})-f_{syn}(x^{(j)},\hat{t},\hat{d})\right\|^2$ است. بنابراین، τ_{pos} خیلی بیشتر از τ_{Syn} است. نشان داده شده است که به طور متوسط τ_{pos} حدود ۱۰ برابر بزرگتر از τ_{Syn} است. این بدین معنی است که ویژگی‌های اجزای سخن به نسبت ویژگی‌های نحوی در گام‌های بزرگتری به‌روز رسانی می‌شود. در نتیجه، ویژگی‌های اجزای سخن نقش مهم‌تری را

در مدل‌های توأم بازی می‌کنند. در نتیجه امتیازهای اجزای سخن در بهترین نتایج توأم به امتیاز نحوی بسیار نزدیک‌ترند.

به دلیل قدرت تمایز ویژگی‌های اجزای سخن در حل ابهامات اجزای سخن که به نحو حساس نیستند مدل‌های توأم زمانی که با AP و PA آموزش داده شده‌اند سرکوب می‌شوند. در مقایسه با PA و AP، SPA وزن ویژگی‌های اجزای سخن را بالا می‌برد و می‌تواند به صورت بهتری از قدرت تمییز ویژگی‌های اجزای سخن و ویژگی‌های نحوی استفاده کند و منجر به تقویت زیاد دقت برچسب‌زنی می‌شود. به عبارت دیگر، نتایج برچسب‌زنی بهتر بیشتر می‌تواند به کمک تجزیه برسد.

۳-۳-٤- مدل توأم مبتنی بر گراف با استفاده از تجزیه‌ی دوگان و گرادیان کاهشی

در روش‌های مبتنی بر گراف با استفاده از تجزیه‌ی دوگان که یکی از مسائل بهینه‌سازی است به صورت گردش تکرار این کار به صورت توأم صورت می‌گیرد. نخست مسأله‌ی برچسب‌زنی و تجزیه‌ی نحوی به صورت جداگانه حل می‌شود. سپس بر اساس محدودیتی مانند این که باید برچسب اجزای سخن موجود در تجزیه‌گر و برچسب‌زن یکی باشد، در زمان آزمون (و نه یادگیری) آن قدر از روش‌های کاهش شیب استفاده می‌شود تا این محدودیت ارضا شود [۲۵].

در ابتدا برای آشنایی با تجزیه‌ی دوگان مقدماتی در مورد آرام‌سازی لاگرانژی ارائه شده و مفهوم تجزیه‌ی دوگان شرح داده می‌شود سپس به بررسی نحوه‌ی یکپارچه‌سازی برچسب‌زنی و تجزیه می‌پردازیم. یکی از شرایط ارضای فرض‌های ریاضی این روش این است که درخت ت جزیه‌ی استفاده شده WCFG باشد اما در [۲۵] آمده است که می‌توان از این روش برای دیگر روش‌های تجزیه نیز بهره برد. در همین راستا با وجود اینکه این سمینار به بررسی مدل‌های توأم برچسب‌زنی و تجزیه‌ی وابستگی می‌پردازد، به دلیل نو بودن و رویکرد متفاوت این روش در سمینار آورده شده است.

۳-۳-٤-۱- تجزیه‌ی دوگان و آرام‌سازی لاگرانژی

تجزیه‌ی دوگان و به طور کلی آرام‌سازی لاگرانژی یک روش کلاسیک برای بهینه‌سازی ترکیبی است. موضوع اصلی آرام‌سازی لاگرانژی به طور طبیعی در رابطه با کلاس گسترده‌ای از الگوریتم‌های ترکیبی است [۲۵].

۳-۳-٤-۲- شرح بحث

در بسیاری از مسائل در پردازش آماری زبان طبیعی وظیفه نگاشت کردن چند ورودی x (مانند یک رشته) به چند خروجی ساخت‌یافته y (مانند یک درخت تجزیه) است. این نگاشت کردن عموما به صورت زیر تعریف می‌شود:

$$y^* = arg \max_{y \in Y} h(y) \qquad (10)$$

که Y مجموعه‌ی متناهی از ساختارهای ممکن برای ورودی x است، و $h:Y \rightarrow R$ تابعی است که امتیاز h(y) را به هر y در Y نسبت می‌دهد. برای مثال، در برچسب‌زنی اجزای سخن، x یک رشته خواهد بود و Y هم مجموعه‌ای از دنباله‌های برچسب ممکن برای x است؛ در تجزیه، x یک رشته است و Y مجموعه‌ای از همه‌ی درخت‌های تجزیه برای x است؛ در ترجمه‌ی ماشینی، x یک رشته‌ی زبان مبدا است و Y مجموعه‌ای از همه‌ی ترجمه‌های ممکن برای x است. مسئله‌ی پیدا کردن y^* به عنوان مسئله‌ی رمزگشایی در نظر گرفته شده است. اندازه‌ی Y عموما به صورت نمایی با در نظر گرفتن اندازه‌ی ورودی x رشد می‌کند و جستجوی جامعی برای به‌دست آوردن y^* مقاوم انجام می‌دهد [۲۵].

تجزیه‌ی دوگان سبب می‌شود که مشاهدات در بسیاری از مسائل رمزگشایی می‌تواند به دو یا چندین زیر مسئله همراه با محدودیت‌های خطی که مفاهیم توافق بین راه حل‌ها برای مسائل مختلف را ایجاد می‌کند، تجزیه می‌شود. زیر مسائل به گونه‌ای انتخاب می‌شوند که با استفاده از الگوریتم‌های ترکیبی دقیق به طور موثری حل می‌شوند. محدودیت‌های توافق با استفاده از ضرایب لاگرانژ گنجانیده می‌شوند و یک الگوریتم تکراری- برای مثال، یک الگوریتم زیر گرادیان- برای کمینه کردن دوگان به‌دست آمده استفاده می‌شود. الگوریتم‌های تجزیه‌ی دوگان دارای ویژگی‌های زیر می‌باشند:

آن‌ها معمولا ساده و کارامد هستند. برای مثال، الگوریتم‌های زیرگرادیان در هر تکرار شامل دو گام‌اند: اول اینکه هر زیر مسئله با یک الگوریتم ترکیبی حل می‌شود؛ و دوم، به روز رسانی افزایشی ساده برای ضرایب لاگرانژی ایجاد می‌شود.

تجزیه‌ی دوگان که در آن از دو یا تعداد بیشتری الگوریتم‌های ترکیبی استفاده می‌شود، یک مورد خاص از آرام‌سازی لاگرانژی (LR) است.

۳-۳-٤-۳- آرام‌سازی لاگرانژی

در مرجع [۲۵] آمده است: مجموعه‌ی متناهی Y که زیر مجموعه‌ای از R^d است در دسترس است. و امتیاز در نظر گرفته شده برای هر بردار $y \in Y$ به صورت زیر است:

$$h(y) = y.\theta$$

که θ هم یک بردار در R^d است. مسئله‌ی رمزگشایی هم پیدا کردن عبارت زیر است:

$$y^* = \operatorname{argmax}_{y \in Y} h(y) = arg \max_{y \in Y} y.\theta \qquad (11)$$

تحت این تعاریف، هر ساختار y به عنوان یک بردار d بعدی نمایش داده می‌شود، و امتیاز برای ساختار y یک تابع خطی است مانند $y.\theta$. در عمل، در مسائل پیش‌بینی سازمان‌یافته y اکثر اوقات یک بردار دودویی است (یعنی $y \in \{0,1\}^d$) و مجموعه قسمت‌های موجود در y را نشان می‌دهد. سپس بردار θ به هر قسمت امتیاز نسبت می‌دهد، و تعریف $h(y)= y.\theta$ بیان می‌کند که امتیاز برای y مجموع امتیازهای قسمت‌هایی است که شامل آن می‌شود.

اولین گام کلیدی در آرام‌سازی لاگرانژی انتخاب یک مجموعه‌ی متناهی $Y' \subset R^d$ است که دارای ویژگی‌های زیر است:

- $Y \subset Y'$. بنابراین Y' شامل همه‌ی بردارهایی است که در Y پیدا می‌شود، و بعلاوه شامل چندین برداری است که در Y نیست.
- برای هر مقدار θ، می‌توانیم به سادگی عبارت زیر را پیدا کنیم: $arg\max_{y \in Y'} y.\theta$

(توجه کنید که در معادله‌ی ۱۱ ، Y را با مجموعه‌ی بزرگتر Y' جایگزین کردیم.) یعنی این مسئله به صورت چشمگیری ساده‌تر از مسئله‌ی معادله‌ی ۱۱ است. برای مثال، مسئله‌ی معادله‌ی ۱۱ ممکن است NP-hard باشد، در حالی که مسئله‌ی جدید در زمان چندجمله‌ای قابل حل است؛ یا اینکه هر دو مسئله در زمان چند جمله‌ای قابل حل‌اند، اما پیچیدگی مسئله‌ی جدید به طور قابل ملاحظه‌ای کمتر است.

فرض می‌کنیم که:

$$Y = \{y: y \in Y' \; and \; Ay = b\} \qquad (12)$$

برای بعضی از $A \in R^{p \times d}$ و $b \in R^p$. شرط $Ay=b$ بیان می‌کند که p روی y محدودیت خطی دارد. فرض می‌کنیم که تعداد محدودیت‌ها، p ، چندجمله‌ای در اندازه‌ی ورودی است.

لازم است تا محدودیت $Ay=b$ به مجموعه‌ی Y' افزوده شود، اما این محدودیت‌ها به طور قابل ملاحظه‌ای مسئله‌ی رمزگشایی را پیچیده می‌کنند. به جای ترکیب آن‌ها به عنوان محدودیت سخت، این محدودیت‌ها با استفاده از آرام‌سازی لاگرانژی مورد بررسی قرار می‌گیرد. بردار ضرایب لاگرانژ معرفی می‌شود، $u \in R^p$. لاگرانژ به صورت زیر است:

$$L(u, y) = y.\theta + u.(Ay - b)$$

این تابع، تابع هدف اصلی $y.\theta$ را با عبارت دوم که یک محدودیت خطی را با ضرایب لاگرانژ یکی می‌کند ترکیب می‌کند. هدف دوگان به صورت زیر است:

$$L(u) = \max_{y \in Y'} L(u, y)$$

و مسئله‌ی دوگان پیدا کردن این عبارت است: $\min_{u \in R^p} L(u)$

یک رویکرد مشترک که در اینجا در تمام الگوریتم‌ها مورد استفاده قرار می‌گیرد استفاده از الگوریتم زیرگرادیان برای کاهش دوگان است. مقادیر اولیه‌ی ضرایب لاگرانژ $u^{(0)} = 0$ قرار داده می‌شود. سپس برای k=1,2,... مراحل زیر اجرا می‌شود.

$$y^{(k)} = arg\max_{y\in Y'} L(u^{(k-1)}, y) \qquad (13)$$

و در ادامه:

$$u^{k} = u^{(k-1)} - \delta_k\left(Ay^{(k)} - b\right) \qquad (14)$$

که $\delta_k > 0$ اندازه‌ی گام‌ها در تکرار kام است. بنابراین در هر تکرار ابتدا ساختار $y^{(k)}$ را به‌دست می‌آوریم، سپس ضرایب لاگرانژ به روز می‌شود که به $y^{(k)}$ بستگی دارد.

$$arg\max_{y\in Y'} L(u^{(k-1)}, y) = arg\max_{y\in Y'} L(y.\theta + u^{(k-1)}.(Ay - b))$$
$$= arg\max_{y\in Y'} y.\theta'$$

که $$\theta' = \theta + A^T u^{(k-1)}$$

بنابراین عبارات ضرایب لاگرانژ به سادگی در تابع هدف ترکیب شده است.

تئوری ۱: ویژگی‌های زیر برای آرام‌سازی لاگرانژی در نظر گرفته می‌شود:

a- برای هر $u \in R^p$، $L(u) \geq \max_{y\in Y} h(y)$.

b- با انتخاب مناسب اندازه‌های گام δ_k، $\lim_{k\to\infty} L\left(u^{(k)}\right) = \min_u L(u)$.

c- اگر $\exists u$ که $Ay_u = b$، سپس $y_u = arg\max_{y\in Y} y.\theta$ (یعنی $y_u = arg\max_{y\in Y'} L(u,y)$ بهینه است). به‌ویژه در الگوریتم زیر گرادیان اگر برای هر kای $\mathrm{A}y^{(k)} = b$ بنابراین $y^{(k)} = arg\max_{y\in Y} y.\theta$.

d- $\min_u L(u) = \max_{\mu\epsilon Q} \mu.\theta$ که مجموعه‌ی Q در زیر تعریف شده است.

بخش a تئوری بیان می‌کند که یک حد بالا روی امتیاز برای جواب بهینه تهیه می‌کند، و بخش b می‌گوید که روش زیرگرادیان با موفقیت این حد بالا را کمینه می‌کند. بخش c بیان می‌کند که هر پاسخ $y^{(k)}$ محدودیت‌های خطی را ارضا می‌کند، پس مسئله‌ی بهینه‌سازی اصلی حل می‌شود.

بخش d تئوری ارتباط مستقیمی بین روش آرام‌سازی لاگرانژی و آرام‌سازی LP مسئله‌ی معادله‌ی ۱۱ ارائه داده است. مجموعه‌ی Q تعریف می‌شود. اول، Δ مجموعه‌ی همه‌ی توزیع‌های روی مجموعه‌ی Y' تعریف می‌شود:

$$\Delta=\left\{\alpha: \alpha \in R^{|Y'|}, \sum_{y\in Y'} \alpha_y = 1, \forall y\ 0 \le \alpha_y \le 1\right\}$$

و پوشش محدب Y' به صورت زیر تعریف می‌شود:

$$conv(Y')=\left\{\mu \in R^d : \exists \alpha \in \Delta\ s.t.\ \mu = \sum_{y\in Y'} \alpha_y y\right\}$$

و مجموعه‌ی Q را به صورت زیر تعریف می‌شود:

$$Q=\{y : y \in Conv(Y')\ and\ Ay = b\}$$

در معادله‌ی ۱۲، Y' را با پوشش محدب Y' جایگزین کردیم. Y' زیرمجموعه‌ی $\mathrm{Conv}(Y')$ است، و بنابراین Y زیرمجموعه‌ی Q است. مسئله $\max_{\mu\in Q} \mu.\theta$ یک برنامه‌ی خطی است، و یک آرام‌سازی از مسئله‌ی اصلی ماست، $\max_{y\in Y} y.\theta$

بخش (d) تئوری ۱ نتیجه‌ی مستقیم دوگان در برنامه‌نویسی خطی است. و نتایج و پیامدهای زیر را دارد:

- با کمینه کردن دوگان $L(u)$، مقدار بهینه‌ی $\max_{\mu\in Q} \mu.\theta$ در آرام‌سازی LP به دست می‌آید.
- اگر $\max_{\mu\in Q} \mu.\theta = \max_{y\in Y} y.\theta$ باشد، آرام‌سازی LP مقید است. در این مورد الگوریتم زیرگرادیان تضمین می‌کند تا جواب بهینه‌ی مسئله‌ی رمزگشایی اصلی پیدا شود.

$$y^* = arg \max_{\mu\in Q} \mu.\theta = arg \max_{y\in Y} y.\theta$$

- در مواردی که آرام‌سازی LP مقید نیست روش‌هایی (برای مثال، Nedic and Ozdaglar (2009) را ببینید.) وجود دارد که با آن‌ها می‌توان جواب تقریبی برنامه‌ی خطی را به‌دست آورد،

$\mu^* = arg \max_{\mu \in Q} \mu.\theta$. متناوبا، از روش‌هایی برای مقیدسازی آرام‌سازی تا به‌دست آوردن جواب دقیق استفاده می‌شود.

۳-۳-٤-٤-تجزیه‌ی دوگان

در مرجع [۲۵] آمده است: تجزیه‌ی دوگان مورد خاصی از آرام‌سازی لاگرانژی است. فرض می‌شود که مجموعه‌ی متناهی $Y \subset R^d$ در اختیار است. هر بردار $y \in Y$ دارای یک امتیاز است.$= f(y)$ $y.\theta^{(1)}$ که $\theta^{(1)}$ برداری در R^d است. به علاوه، مجموعه‌ی متناهی دوم $Z \subset R^{d'}$ نیز فرض می‌شود که برای هر بردار $z \in Z$ یک امتیاز به دست می‌آید:

$$g(z) = z.\theta^{(2)}$$

سپس مسئله‌ی رمزگشایی پیدا کردن عبارت زیر است.

$$arg \max_{y \in Y, z \in Z} y.\theta^{(1)} + z.\theta^{(2)}$$

که Ay+Cz=b

که $A \in R^{p \times d}$ و $C \in R^{p \times d'}$ و $b \in R^p$.

بنابراین مسئله‌ی رمزگشایی، مسئله‌ی پیدا کردن زوج ساختارهای بهینه بر اساس محدودیت خطی Ay+Cz=b است. در عمل، محدودیت‌های خطی اکثر اوقات محدودیت‌های توافقی بین y و z است: برای همین این دو بردار در بعضی جهات منسجم هستند. برای سادگی، و ایجاد ارتباط واضح‌تری با آرام‌سازی لاگرانژی، مجموعه‌های زیر تعریف می‌شود:

$$w = \{(y,z): y \in Y, z \in Z, Ay + Cz = b\}$$

$$w' = \{(y,z): y \in Y, z \in Z\}$$

بنابراین مسئله‌ی رمزگشایی پیدا کردن عبارت زیر است:

$$arg \max_{(y,z) \in w} (y.\theta^{(1)} + z.\theta^{(2)}) \qquad (15)$$

در مرحله‌ی بعدی فرض‌های زیر در نظر گرفته می‌شود:

- برای هر مقدار $\theta^{(1)} \epsilon R^d$ به آسانی می‌توان مقدار $arg \max_{y \in Y} y.\theta^{(1)}$ را پیدا کرد. به علاوه، برای هر مقدار $\theta^{(2)} \epsilon R^{d'}$ ، به سادگی می‌توان مقدار $arg \max_{z \in Z} z.\theta^{(2)}$ را یافت. در ادامه برای هر $\theta^{(1)} \epsilon R^d$ و $\theta^{(2)} \epsilon R^{d'}$ به می‌توان عبارت زیر را پیدا کرد.
- $$(y^*, z^*) = arg \max_{(y,z) \in w'} y.\theta^{(1)} + z.\theta^{(2)} \qquad (16)$$

با در نظر گرفتن:

$$y^* = arg \max_{y \in Y} y.\theta^{(1)}, \qquad z^* = arg \max_{z \in Z} z.\theta^{(2)}$$

فرض کنید که معادله‌ی ۱۶ با جابه‌جا کردن W با w' ارتباط نزدیکی با مسئله‌ی معادله‌ی ۱۵ دارد. (یعنی محدودیت‌های خطی Ay+Cz=b کم شده است). این مسائل بهینه‌سازی به طور قابل ملاحظه‌ای آسانتر از مسئله‌ی اصلی معادله‌ی ۱۵ حل می‌شوند. هدف بهینه‌سازی یک هدف خطی بر روی مجموعه‌ی متناهی W است همانی که در معادله‌ی ۱۵ داده شده است؛ می‌توان به صورت کارایی مقدار بهینه را روی مجموعه‌ی W' به‌دست آورد زیرا W زیرمجموعه‌ی W' است و W' محدودیت‌های خطی Ay+Cz=b را کاهش داده است. الگوریتم تجزیه‌ی دوگان هم مشابه روش‌هایی دیده شده ایجاد شده است. یک بردار ضرایب لاگرانژی تعریف می‌شود، $u \in R^p$. لاگرانژ به صورت زیر است:

$$L(u, y, z) = y.\theta^{(1)} + z.\theta^{(2)} + u.(Ay + Cz - b)$$

و هدف دوگان به صورت زیر است:

$$L(u) = \max_{(y,z) \in w'} L(u, y, z)$$

دوباره الگوریتم زیرگرادیان می‌تواند برای یافتن $\min_{u \in R^p} L(u)$ مورد استفاده قرار گیرد. ضرایب لاگرانژ به صورت $u^{(0)} = 0$ مقداردهی اولیه می‌شود. برای k=1,2,... گام‌های زیر اجرا می‌شود:

$$(y^k, z^k) = \arg \max_{(y,z) \in w'} L(u^{(k-1)}, y, z)$$

در ادامه $u^{(k)} = u^{(k-1)} - \delta_k(Ay^{(k)} + Cz^{(k)} - b)$

که $\delta_k > 0$ اندازه‌ی گام است. دقت کنید که $y^{(k)}$ و $z^{(k)}$ در هر تکرار به آسانی پیدا می‌شوند زیرا:

$$arg \max_{(y,z)\in w'} L(u^{(k-1)}, y, z) = (arg \max_{y\in Y} y.\theta'^{(1)}, arg \max_{z\in Z} z.\theta'^{(2)},)$$

که $\theta'^{(2)} =. \theta^{(2)} + C^T u^{(k-1)}$ و $\theta'^{(1)} =. \theta^{(1)} + A^T u^{(k-1)}$. بنابراین دوگان به دو مسئله‌ی بیشینه‌سازی که به راحتی قابل حل‌اند تفکیک می‌شود. ویژگی‌های رسمی برای تجزیه‌ی دوگان بسیار شبیه آن‌هایی اند که در تئوری ۱ دیده شد. به‌ویژه، می‌توان نشان داد:

$$\min_{u\in R^p} L(u) = \max_{(\mu,v)\in Q} \mu.\theta^{(1)} + v.\theta^{(2)}$$

که مجموعه‌ی Q به صورت زیر تعریف می‌شود.

$$Q=\{(\mu, v): (\mu, v) \in Conv(W') \; and \; A\mu + Cv = d\}$$

صورت مسئله عبارت زیر است:

$$\max_{(\mu,v)\in Q} \mu.\theta^{(1)} + v.\theta^{(2)}$$

این مسئله دوباره یک مسئله‌ی پیاده‌سازی خطی است، و L(u) دوگان این برنامه‌ی خطی است.

۳-۳-٤-۵- یکپارچه‌سازی تجزیه‌گر و یک برچسب‌زن با تعداد حالت متناهی

در این بخش یک الگوریتم تجزیه‌ی دوگان برای رمزگشایی بر اساس یک مدل که گرامر مستقل از متن وزن‌دار را با یک برچسب‌زن با تعداد حالات متناهی ترکیب می‌کند، شرح داده می‌شود. مسئله‌ی نگاشت یک جمله‌ی ورودی x را به درخت تجزیه‌ی y در نظر بگیرید. Y را مجموعه‌ی همه‌ی درخت‌های تجزیه برای x در نظر بگیرید. مسئله‌ی تجزیه پیدا کردن عبارت زیر است.

$$y^* = arg \max_{y\in Y} h(y) \qquad (17)$$

که $h(y)$ امتیاز درخت تجزیه‌ی $y\in Y$ است.

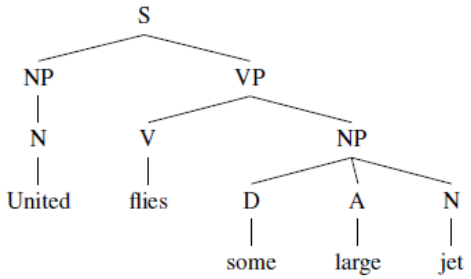
موردی را در نظر بگیرید که $h(y)$ جمع امتیازهای دو مدل است: اول، امتیاز y تحت یک گرامر مستقل از متن وزن‌دار؛ و دوم، امتیاز برای دنباله‌ی اجزای سخن(اجزای سخن) در y تحت یک مدل برچسب‌زنی اجزای سخن با تعداد حالات متناهی. به صورت رسمی‌تر h(y) به صورت زیر تعریف می‌شود.

$$h(y)=f(y)+g(l(y)) \qquad (18)$$

که تابع‌های f،g و l به صورت زیر تعریف می‌شود.

۱- f(y) امتیاز y تحت یک گرامر مستقل از متن وزن‌دار (WCFG) است. یک WCFG شامل یک گرامر مستقل از متن با مجموعه‌ای از قواعد G است، و تابع امتیاز: $\theta: G \to R$ است که یک امتیاز با مقدار حقیقی را به هر قاعده در G نسبت می‌دهد. امتیاز یک درخت تجزیه‌ی کامل مجموع امتیازهای قواعدی است که شامل آن است. برای مثال، درخت تجزیه‌ای را که در شکل ۱۷ زیر نشان داده شده است در نظر بگیرید؛ برای این درخت،

$$f(y) = \theta(S \to NP\ VP) + \theta(NP \to N) + \theta(N \to United) + \theta(VP \to V\ NP) + \cdots$$



شکل ۱۷- یک مثال از درخت تجزیه [۲۵]

. در گرامر مستقل از متن احتمالی $\theta(\alpha \to \beta) = \log p(\alpha \to \beta|\alpha)$ تعریف می‌شود. در مثالی دیگر، در میدان تصادفی شرطی(CRF) به صورت $\theta(\alpha \to \beta) = w.\emptyset(\alpha \to \beta)$ در نظر گرفته می‌شود که $w \in R^q$ یک بردار پارامتری است، و $\emptyset(\alpha \to \beta) \in R^q$ یک بردار ویژگی است که قاعده‌ی $\alpha \to \beta$ را نمایش می‌دهد.

۲- l(y) تابعی است که درخت تجزیه‌ی y را به دنباله‌ی برچسب‌های اجزای سخن در y نگاشت می‌کند. برای درخت تجزیه‌ی شکل ۱۷، l(y) دنباله‌ی N V D A N را می‌دهد.

۳- $g(z)$ امتیاز دنباله‌ی برچسب اجزای سخن z تحت مدل برچسب‌زنی با تعداد حالات متناهی از مرتبه‌ی m است. در این مدل، اگر z_i برای $i=1...n$ برچسب iام در z باشد، سپس

$$g(z) = \sum_{i=1}^{n} \theta(i, z_{i-m}, z_{i-m+1}, \dots, z_i)$$

که $\theta(i, z_{i-m}, z_{i-m+1}, \dots, z_i)$ امتیاز برای زیردنباله‌ای با برچسب‌های $z_{i-m}, z_{i-m+1}, \dots, z_i$ است که در موقعیت iام دنباله تمام می‌شود.[۵۵] هنوز مشخص نیست که عبارت θ چگونه تعریف می‌شود. به عنوان مثال، g(z) می‌تواند احتمال لگاریتمی برای z تحت مدل مارکوف پنهانی باشد. که در این مورد

$$\theta(i, z_{i-m} \dots z_i) = \log p(z_i | z_{i-m} \dots z_{i-1}) + \log p(x_i | z_i)$$

که x_i کلمه‌ی iام در دنباله‌ی ورودی است. به عنوان یک مثال دیگر، تحت CRF داریم

$$\theta(i, z_{i-m} \dots z_i) = w.\emptyset(x, i, z_{i-m} \dots z_i)$$

که $w \in R^q$ یک بردار پارامتری است و $\emptyset(x, i, z_{i-m} \dots z_i)$ نمایش بردار ویژگی زیردنباله‌ی برچسب‌های $z_{i-m} \dots z_i$ است که در موقعیت iام در دنباله‌ی x خاتمه پیدا می‌کند.

تابع امتیازدهی h(y)=f(y)+g(l(y)) اطلاعات را از مدل تجزیه و مدل برچسب‌زنی ترکیب می‌کند. این دو مدل اساسا دو نوع متفاوت از اطلاعات را به‌دست می‌آورند: به ویژه، برچسب‌زن اجزای سخن اطلاعاتی راجع به برچسب‌های اجزای سخن مجاور که ممکن است در f(y) در نظر گرفته نشده باشند را به‌دست می‌آورد. این اطلاعات در مقایسه با f(y) به تنهایی هم کارایی تجزیه و هم برچسب‌زنی را بالا می‌برد.

---

[۵۵] - z_i را برای $i \leq 0$ به عنوان نماد اجزاي سخن شروع تعريف مي‌شود.

بر اساس این تعریف ***h(y)***، رویکرد مرسوم برای پیدا کردن y^* در معادله‌ی ۱۷ ساختن یک گرامر مستقل از متن جدید است که حساسیت به بایگرام‌های سطحی را نشان می‌دهد. به طور کلی در این روش قواعدی نظیر $VP \rightarrow V\ NP$ با قواعدی نظیر

$$VP_{N,N} \rightarrow V_{N,V}\ NP_{V,N} \qquad (19)$$

جایگزین می‌شود. به طوری که هر غیر ترمینالی(مانند ***NP***) با غیر ترمینالی که برچسب اجزای سخن پیشین و آخری با یک غیر ترمینال دیگر ارتباط دارد جایگزین می‌شود. برای مثال، $NP_{V,N}$ ،***NP*** ای را ارائه می‌کند که بر زیر درختی که بر چسب اجزای سخن پیشین آن ***V*** و آخرین برچسب اجزای سخن آن ***N*** بوده است تسلط دارد. وزن‌های قواعد جدید وزن‌های مستقل از متن ***y*** می‌باشند. به علاوه، قواعدی نظیر $V \rightarrow files$

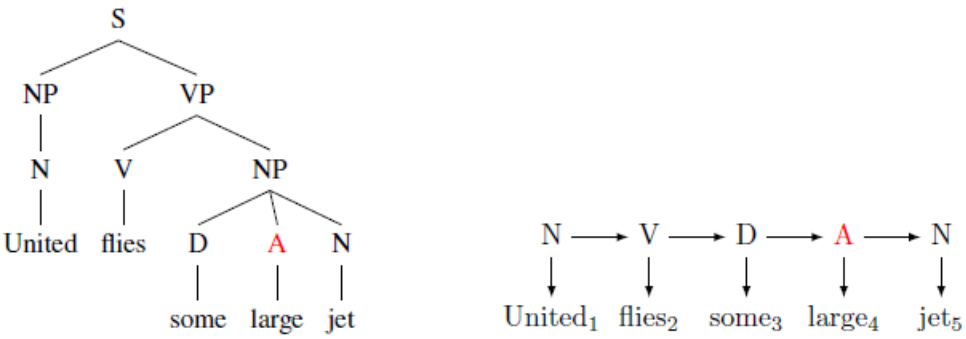
با قواعدی به صورت زیر جابه‌جا می‌شوند. $V_{N,V} \rightarrow files$

وزن‌ها در این قواعد وزن‌های مستقل از متن ***f(y)*** به علاوه وزن‌های برچسب بایگرام ***g(z)*** هستند. در این مثال برای بایگرام ***N V***. الگوریتم تجزیه‌ی برنامه‌نویسی پویا – برای مثال الگوریتم ***CYK***- برای یافتن ساختار بیشترین امتیازدهی در گرامر جدید مورد استفاده قرار می‌گیرد. این روش تضمین می‌کند تا جواب دقیقی برای مسئله‌ی معادله‌ی ۱۷ ارائه شود؛ گرچه در اغلب موارد بسیار ناکارامد است. اندازه‌ی گرامر با معرفی غیر ترمینال‌های پالایش شده افزایش داده می‌شود و این به طور قابل ملاحظه‌ای به سمت کارایی کمتر تجزیه پیش می‌رود. به عنوان مثال، در موردی که گرامر مورد نظر یک ***CFG*** در فرم نرمال چامسکی است، با ***G*** تا غیر ترمینال، و از مدل برچسب‌زنی تریگرام(مرتبه‌ی دوم) با ***T*** تا برچسب اجزای سخن ممکن استفاده می‌شود. ***n*** طول جمله‌ی ورودی در نظر گرفته می‌شود.

۳-۳-٤-٦- الگوریتم تجزیه‌ی دوگان

فرض کنید τ مجموعه‌ی همه‌ی برچسب‌های اجزای سخن باشد. فرض کنید که جمله‌ی ورودی دارای n کلمه است. برای هر درخت تجزیه‌ی y، برای هر موقعیت $i \in \{1 \dots n\}$، برای هر برچسب $t \in \tau$، $y(i,t) = 1$ تعریف می‌شود اگر، درخت تجزیه‌ی y در موقعیت i دارای برچسب t باشد وگرنه

$y(i,t) = 0$. به صورت مشابه، برای هر دنباله‌ی z ، اگر دنباله‌ی برچسب‌ها در موقعیت i بر چسب t داشته باشد $z(i,t) = 1$ را تعریف می‌شود وگرنه، برابر ۰ است. به عنوان مثال، در درخت تجزیه و دنباله‌ی برچسب **زیر**: y(4,A)=1 و z(4,A)=1 [۲۵]:



به علاوه، Z را به عنوان مجموعه‌ی همه‌ی دنباله‌های برچسب اجزای سخن برای جمله‌ی ورودی تعریف می‌شود. سپس مسئله‌ی بهینه‌سازی به صورت زیر تعریف می‌شود:

مسئله‌ی بهینه‌سازی ۱: عبارت زیر را پیدا کنید:

$$arg \max_{y \in Y, z \in Z} f(y) + g(z) \qquad (20)$$

که برای همه‌ی $i \in \{1 \dots n\}$، و برای همه‌ی $t \in T$، y(i,t)=z(i,t).

بنابراین بهترین زوج ساختارهای y و z را می‌توان پیدا کرد که آن‌ها برچسب یکسانی را به اشتراک می‌گذارند. y^* همان argmax مسئله در معادله‌ی ۱۷ است. در این معنا، حل مسئله‌ی جدید به سمت حل مسئله‌ی اصلی هدایت می‌شود. دو فرض زیر در نظر گرفته می‌شود. اینکه آیا این تعاریف ارضا می‌شوند به تعریف y و f(y) (برای فرض ۱) و تعاریف z و g(z) (برای فرض ۲) بستگی دارد. این فرض‌ها زمانی انجام می‌گیرند که f(y) یک WCFG است و g(z) یک برچسب‌زن با تعداد حالت‌های متناهی است، اما به طور کلی ممکن است برای دیگر مدل‌های تجزیه و برچسب‌زنی هم برقرار باشد.

فرض ۱- فرض کنید که متغیرهای بدین صورت تعریف شود، $u(i,t) \in R$ برای $i \in \{1 \dots n\}$ و $t \in \tau$. فرض می‌شود که برای هر مقداری از این متغیرها، می‌توان عبارت زیر را به صورت کارامدی پیدا کرد.

$$arg \max_{y \in Y}(f(y) + \sum_{i,t} u(i,t)y(i,t))$$

یک مثال: فرض کنید که WCFG گرامری به فرم نرمال چامسکی است. تابع امتیازدهی به صورت زیر تعریف شده است.

$$f(y) = \sum_{X \to YZ} c(y, X \to YZ)\theta(X \to YZ) + \sum_{i,t} y(i,t)\theta(t \to w_i)$$

وقتی می‌نویسیم $c(y, X \to YZ)$ منظور تعداد دفعاتی است که قانون $X \to YZ$ در درخت تجزیه‌ی y دیده شده است، و مانند گذشته y(i,t)=1 است اگر کلمه‌ی i دارای برچسب t باشد، وگرنه برابر صفر است(توجه کنید که y(i,t)=1 بیان می‌کند که قانون $t \to w_i$ در درخت تجزیه استفاده شده است). درخت تجزیه با بیشترین امتیاز تحت f(y) می‌تواند به صورت کارامدی پیدا شود، برای مثال با استفاده از الگوریتم تجزیه‌ی cyk. سپس:

$$\begin{aligned} arg \max_{y \in Y}(f(y) + \sum_{i,t} u(i,t)y(i,t)) \\ = arg \max_{y \in Y}(\sum_{X \to YZ} c(y, X \to YZ)\theta(X \to YZ) \\ + \sum_{i,t} y(i,t)(\theta(t \to w_i) + u(i,t))) \end{aligned}$$

این argmax می‌تواند به سادگی با الگوریتم CYK به‌دست آورده شود، که امتیازهای $\theta(t \to w_i)$ به‌سادگی با امتیازهای جدید تعریف شده مانند $\theta'(t \to w_i) = \theta(t \to w_i) + u(i,t)$ جایگزین شود.

فرض ۲- متغیرهای زیر را تعریف کنید $u(i,t) \in R$ برای $i \in \{1 \dots n\}$ و $t \in \tau$. برای هر مقدار این متغیرها، عبارت زیر به صورت کارایی پیدا می‌شود:

$$arg\max_{z\in Z}(g(z)+\sum_{i,t}u(i,t)z(i,t))$$

یک مثال. یک مدل برچسب‌زنی مرتبه‌ی ۱ در نظر بگیرید،

$$g(z)=\sum_{i=1}^{n}\theta(i,z_{i-1},z_i)$$

سپس

$$arg\max_{z\in Z}\left(g(z)-\sum_{i,t}u(i,t)z(i,t)\right)$$
$$=arg\max_{z\in Z}\left(\sum_{i=1}^{n}\theta(i,z_{i-1},z_i)-\sum_{i,t}u(i,t)z(i,t)\right)$$
$$=arg\max_{z\in Z}\sum_{i=1}^{n}\theta'(i,z_{i-1},z_i)$$

که $\theta'(i,z_{i-1},z_i)=\theta(i,z_{i-1},z_i)-u(i,z)$

که argmax می‌تواند به صورت کارایی با استفاده از الگوریتم ویتربی به‌دست آید، که عبارت‌های جدید θ' برای ترکیب کردن و یکپارچه کردن مقادیر u(i,t) مورد استفاده قرار می‌گیرد. با این فرض‌های داده شده، الگوریتم تجزیه‌ی دوگان در شکل ۱۸ نشان داده شده است. این الگوریتم بردار متغیرهای $u=\{u(i,t):i\in\{1\dots n\},t\in\tau\}$ را دستکاری می‌کند. خواهیم دید که هر متغیر $u(i,t)$ یک ضریب

لاگرانژی است که محدودیت y(i,t)=z(i,t) را در مسئله‌ی بهینه‌سازی مورد نظر تحمیل می‌کند. در هر تکرار، الگوریتم فرضیه‌های y^k و z^k را پیدا می‌کند؛ با فرض‌های ۱ و ۲ این گام کارا است.

Initialization: Set $u^{(0)}(i,t) = 0$ for all $i \in \{1 \dots n\}, t \in \mathcal{T}$

For $k = 1$ **to** K

$y^{(k)} \leftarrow \text{argmax}_{y \in \mathcal{Y}} \left(f(y) + \sum_{i,t} u^{(k-1)}(i,t) y(i,t) \right)$ [Parsing]

$z^{(k)} \leftarrow \text{argmax}_{z \in \mathcal{Z}} \left(g(z) - \sum_{i,t} u^{(k-1)}(i,t) z(i,t) \right)$ [Tagging]

If $y^{(k)}(i,t) = z^{(k)}(i,t)$ for all i,t **Return** $(y^{(k)}, z^{(k)})$

Else $u^{(k+1)}(i,t) \leftarrow u^{(k)}(i,t) - \delta_k (y^{(k)}(i,t) - z^{(k)}(i,t))$

شکل ۱۸- الگوریتم تجزیه‌ی دوگان برای تجزیه و برچسب‌زنی یکپارچه. $\boldsymbol{\delta_k}$ برای k=1...K اندازه‌ی گام در تکرار kام است .[۲۵]

اگر دو ساختار دنباله‌ی اجزای سخن یکسان داشته باشند (یعنی، $y^k(i,t) = z^{(k)}(i,t)$ برای همه‌ی (i,t) ها) الگوریتم پاسخ را بر می‌گرداند. وگرنه، به‌روز آوری‌های ساده بر روی متغیرهای u(i,t) بر اساس مقادیر $y^k(i,t)$ و $z^k(i,t)$ صورت می‌گیرد.

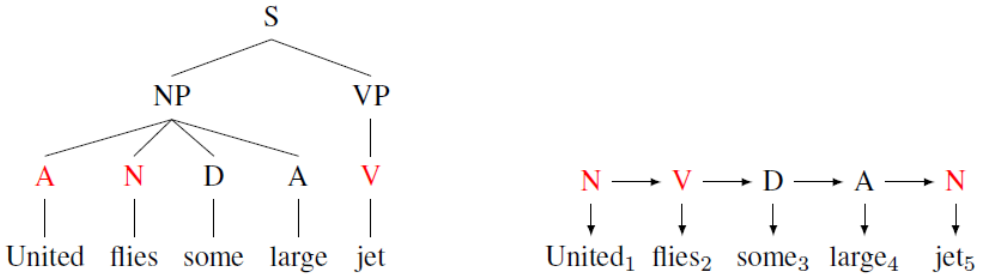
مثالی ارائه می‌شود که الگوریتم را اجرا می‌کند. در ابتدا یک تئوری مهم ارائه می‌شود.

تئوری ۲- اگر در هر تکرار از الگوریتم در شکل ۱۸ برای همه‌ی (i,t) ها داشته باشیم $y^k(i,t) = z^k(i,t)$، سپس، $(y^{(k)}, z^{(k)})$ جواب مسئله‌ی بهینه‌سازی ۱ است. این تئوری نتیجه‌ی مستقیم تئوری ۵ است. در گام بعدی، کارایی الگوریتم را در نظر بگیرید. موردی را در نظر بگیرید که f(y) یک CFG وزن‌دار تعریف شده است، و g(z) یک برچسب‌زن با تعداد حالت‌های متناهی است. هر تکرار الگوریتم به رمزگشایی تحت نظر این دو مدل احتیاج دارد. همان‌طور که قبلا بحث شد، فرض گرامر مستقل از متن در فرم نرمال چامسکی و برچسب‌زن تریگرام با T برچسب در الگوریتم CYK به زمان $O(G^3 n^3)$ و الگوریتم ویتربی برای برچسب‌زنی به زمان $O(T^3 n)$ احتیاج دارد. بنابراین، زمان کلی اجرا برای

الگوریتم تجزیه‌ی دوگان $O(k(G^3n^3 + T^3n))$ است. نتایج الگوریتم تجزیه‌ی دوگان یک هزینه‌ی افزایشی برای ترکیب کردن و متحد کردن برچسب‌زن دارد (یک عبارت T^3n به زمان اجرا اضافه می‌شود.

۳-۲-٦-۵- مثالی از اجرای الگوریتم

در مرجع [۲۵] آمده است: مثالی از اجرای الگوریتم ارائه شده است. برای سادگی، فرض می‌شود که اندازه‌ی گام δ_k برای همه‌ی تکرارهای k برابر 1 است. جمله‌ی ورودی United flies some large jet در نظر گرفته می‌شود. در آغاز مجموعه‌های الگوریتم برای همه‌ی (i,t) ها y(i,t)=0 است. برای مثال رمزگشایی با این وزن‌های اولیه به سوی دو فرضیه هدایت می‌شود.

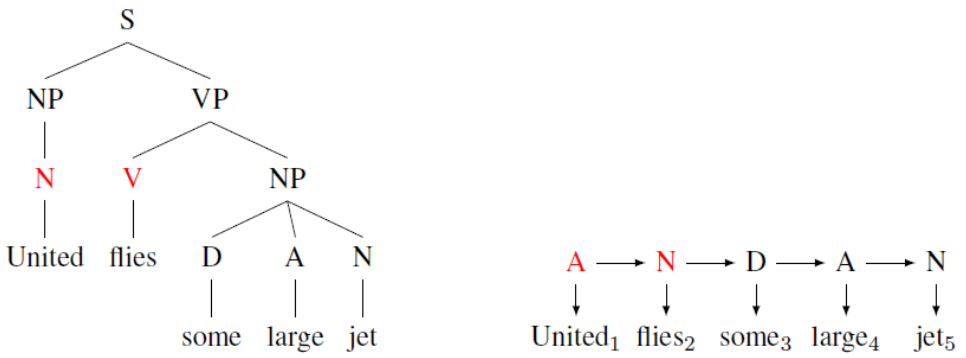


این دو ساختار در سه موقعیت دارای برچسب‌های اجزای سخن متفاوتی هستند که با رنگ قرمز نشان داده شده است؛ بنابراین، دو ساختار با یکدیگر موافق نیستند. سپس متغیرهای u(i,t) را براساس این تفاوت‌های به‌روزرسانی می‌کنیم، مقادیر جدید به صورت زیراند:

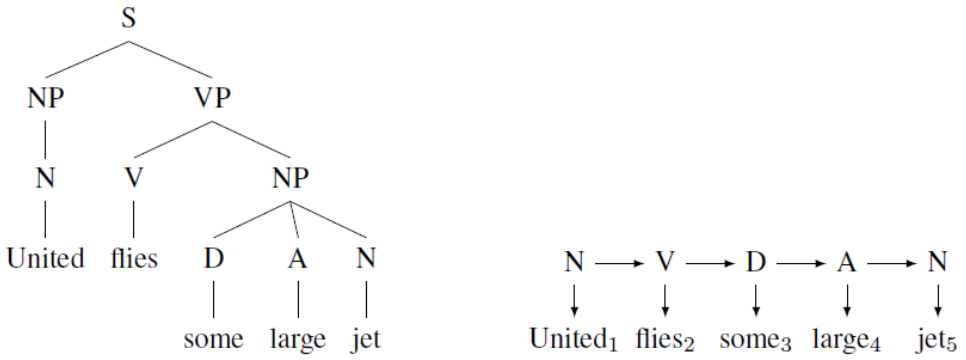
u(1,A)=u(2,N)=u(5,V)=-1 و u(1,N)=u(2,V)=u(5,V)=1

هر مقدار u(i,t) هم که نشان داده نشده است دارای مقدار *0* است. هم‌اکنون با مقادیر جدید u(i,t) رمزگشایی می‌کنیم، ساختارها به صورت زیر داده شده است:

دوباره تفاوت بین ساختارها به رنگ قرمز نشان داده شده است. مقادیر u(i,t) به‌روز رسانی می‌شود تا مقادیر زیر را به دست آید:



u(5,N)=-1 و u(5,v)=1 و بقیه‌ی مقادیر u(i,t)=0 است. (توجه کنید که به روز رسانی u(1,A), u(1,N), u(2,N) را ریست می‌کند و u(2,V) را به صفر بر می‌گرداند.) با مقادیر u(i,t) جدید دوباره رمزگشایی انجام می‌شود؛ دو ساختار به صورت زیر اند.



این دو ساختار دنباله‌های یکسانی از برچسب‌های اجزای سخن دارند، و خاتمه‌ی الگوریتم تضمین می‌کند که پاسخ‌ها بهینه‌اند.

۳-٤- مدل‌های توأم ترکیبی

۳-٤-۱- مدل توأم پشته‌ای

مدل‌های توأم قبلی برچسب‌زنی اجزای سخن و تجزیه‌ی وابستگی از مدل‌های وابستگی مبتنی بر گراف و مبتنی بر گذار گسترش یافته است. تحلیل‌ها نشان می‌دهد که این دو مدل توزیع‌های خطای متفاوتی دارند. به علاوه، یکپارچه‌سازی تجزیه‌ی وابستگی مبتنی بر گذار و مبتنی بر گراف با یادگیری پشته‌ای(پشته‌سازی) به بهبودهای قابل ملاحظه‌ای دست پیدا کرده است. این دلیل خوبی برای مطالعه‌ی مسئله‌ی پشته‌سازی مدل‌های توأم مبتنی بر گراف و مبتنی بر گذار است. [۱۱].

در مرجع [۱۱] آمده است: [۲۶] نشان می‌دهد که کارایی تجزیه‌ی وابستگی می‌تواند با پشته‌سازی یک تجزیه‌گر وابستگی مبتنی بر گراف و یک تجزیه‌گر وابستگی مبتنی بر گذار به میزان بسیار زیادی بهبود پیدا کند. بنابراین بررسی تأثیر یادگیری پشته‌ای زمانی که به مدل‌های توأم اعمال می‌شود جالب است.
مدل‌های توأم مبتنی بر گراف و مبتنی بر گذار به ترتیب گسترش یافته‌ی مدل‌های تجزیه‌ی وابستگی مبتنی بر گراف و مبتنی بر گذار هستند. آن‌ها دو روش اصلی برای تجزیه‌ی وابستگی‌اند. تجزیه‌گرهای دستور مستقل از متن احتمالی(PCFG) مانند تجزیه‌گر Brown [27] و تجزیه‌گر Berkeley [۲۸, ۲۹] که قبلا برای تجزیه‌ی سازه مورد استفاده قرار می‌گرفتند و همچنین برای تجزیه‌ی وابستگی نیز پیشنهاد شده‌اند[۳۰, ۳۱]. این روش‌ها با مدل‌های مبتنی بر سازه مشخص می‌شود. پیش‌شرط‌های مدل‌های مبتنی بر سازه این است که ساختار سازه‌ی خروجی تجزیه‌گرهای PCFG می‌تواند با قواعد کافی به ساختار وابستگی منتقل شود. به علاوه، مدل‌های PCFG می‌توانند برچسب‌زنی اجزای سخن را به صورت همزمان در تجزیه‌ی سازه‌ای پردازش کند. آن‌ها با برچسب‌زنی اجزای سخن به عنوان یک زیر ماژول از تجزیه‌ی سازه‌ای رفتار می‌کنند. بنابراین می‌توان یک مدل PCFG را برای برچسب‌زنی اجزای سخن و تجزیه‌ی وابستگی در نظر گرفت.این با مدل توأم مبتنی بر سازه مشخص می‌شود.
در این بخش، ابتدا یکپارچه‌سازی یک مدل توأم مبتنی بر گراف(JGraph) و یک مدل توأم مبتنی بر گذار (JTrans) با یادگیری پشته‌ای مطالعه می‌شود. یادگیری پشته‌ای، با استفاده از معماری دو سطحی پیاده‌سازی می‌شود که شامل یک یا تعداد بیشتری پیش‌بینی کننده است که نتایج آن‌ها، به عنوان ورودی برای تقویت پیش‌بینی کننده‌ی سطح ۱ مورد بهره‌برداری قرار می‌گیرد. بنابراین JGraph و JTrans می‌تواند به عنوان مدل سطح ۱ مورد استفاده قرار بگیرد. مدل پشته‌ای با استفاده از JGraph با استفاده از مدل سطح اول به وسیله‌ی مدل توأم مبتنی بر گراف ذکر شده فراخوانی می‌شود و مدل پشته‌ای از JTrans به عنوان مدل سطح ۱ به وسیله‌ی مدل توأم مبتنی بر گذار داده شده استفاده می‌کند.
در این بخش، دو استراتژی برای JGraph و JTrans پشته‌ای وجود دارد. مدل توأم مبتنی بر گراف داده شده از JGraph به عنوان مدل سطح یک بهره‌برداری می‌کند و مدل توأم مبتنی بر گذار داده شده از JTrans به عنوان مدل سطح یک استفاده می‌کند.

۳-٤-۱-۱-مدل توأم مبتنی بر گراف راهنما

مدل توأم مبتنی بر گراف، JGraph(JTrans) نامیده می‌شود و از JGraph به عنوان مدل سطح یک و از JTrans به عنوان مدل سطح صفر استفاده می‌شود. مدل توأم مبتنی بر گراف یک درخت وابستگی

همراه با برچسب‌های اجزای سخن با در نظر گرفتن بخش‌های کوچک شامل وابستگی‌ها، برادری‌ها و نوه‌ها محاسبه می‌کند. فرض می‌شود خروجی JTrans، ($\hat{t}^{JTrans} = t_1^{JTrans} \dots t_n^{JTrans}, \hat{d}^{JTrans}$) است، تابع امتیاز JGraph(JTrans) به صورت زیر است:

$$\begin{aligned} Score_{joint}(x,t,d) &= \sum_{\{(h,m)\}\subseteq d} w_{pos} \cdot f_{pos}(x,t,\hat{t}^{JTrans},m) \\ &+ w_{dep} \cdot f_{dep}\left(x,t,h,m,\hat{d}^{JTrans}\right) \\ &+ \sum_{\{(h,s)(h,m)\}\subseteq d} w_{sib} \cdot f_{sib}\left(x,t,h,s,m,\hat{d}^{JTrans}\right) \\ &+ \sum_{\{(g,h)(h,m)\}\subseteq d} w_{grd} \cdot f_{grd}\left(x,t,g,h,m,\hat{d}^{JTrans}\right) \quad (26) \end{aligned}$$

توابع ویژگی ویرایش شده‌اند تا شامل آرگومان اضافی $\hat{t}^{JTrans}$ و $\hat{d}^{JTrans}$ بشود. بنابراین ویژگی‌های جدید مرتبط با آرگومان اضافی تولید می‌شود. این ویژگی‌های جدید برای ویژگی‌های داده شده بر روی خروجی JTrans به حساب می‌آیند [۲۵].

۳-٤-۱-۲- مدل توأم مبتنی بر گذار

مدل توأم مبتنی بر گذار که JTrans(JGraph) نامیده می‌شود، از JTrans به عنوان مدل سطح ۱ و از JGraph به عنوان مدل سطح ۰ بهره‌برداری می‌کند. مدل توأم مبتنی بر گذار یک درخت همراه با وابستگی را همراه با برچسب‌های POS توسط دنباله‌ی اقدام‌های گذار شکل‌دهی آن محاسبه می‌کند. فرض کنیم که خروجی JTrans ($\hat{t}^{JGraph} = t_1^{JGraph} \dots t_n^{JointG}, \hat{d}^{JGraph}$) است و تابع امتیاز JTrans(JGraph) می‌شود:

$$\begin{aligned} Score_{joint}(x,t,d) &= \sum_{A_i = SHIFT(t)} w_{pos} \cdot f_{pos}(ST_i, A_i, t, \hat{t}^{JGraph}) \\ &+ \sum w_{syn} \cdot f_{syn}\left(ST_i, A_i, \hat{d}^{JGraph}\right) \end{aligned}$$

توابع ویژگی ویرایش شده‌اند تا شامل آرگومان‌های اضافی $\hat{t}^{JGraph}$ و $\hat{d}^{JGraph}$ بشوند. بنابراین ویژگی‌های جدید با آرگومان‌های اضافی که تولید می‌شوند در ارتباط‌اند.

۳-۵- نتیجه‌گیری

در این فصل ابتدا به بررسی تعدادی از مدلهای توأم پرداخته شد. سیستمی مبتنی برگذار برای برچسب‌زنی اجزای سخن و تجزیه‌ی وابستگی برچسب خورده با درخت‌های غیر افکنشی بررسی شد و در ادامه یک نوع الگوریتم یادگیری برخط منفعل-پرخاشگر(PA) به نام یک الگوریتم منفعل-پرخاشگر جداگانه مورد بررسی قرار گرفت که به صورت جداگانه وزن‌های ویژگی اجزای سخن و وزن‌های ویژگی نحوی را به‌روزرسانی می‌کند و به طور طبیعی وزن‌های ویژگی‌های اجزای سخن را با استفاده از چارچوب بهینه‌سازی توأم بالا می‌برد. همچنین اولین رویکرد افزایشی ارائه شده برای وظیفه‌ی برچسب‌زنی اجزای سخن و تجزیه‌ی وابستگی پیشنهاد شده بررسی شد که براساس چارچوب تجزیه‌ی جابه‌جایی-کاهش[۵۶] با برنامه‌نویسی پویا ایجاد شده است. همچنین یک مدل توأم برای یکپارچه‌سازی برجسب‌زنی و تجزیه که مبتنی بر تجزیه‌ی دوگان و روش گرادیان کاهشی بود ارائه شد. همچنین یک مدل پشته‌ای مورد بررسی قرار گرفت که مدل‌های مختلف را با یکدیگر ترکیب می‌کرد.

[56] Shift-reduce

فصل چهارم

جمع‌بندی و کارهای آینده

٤- نتیجه گیری و کارهای آینده

٤-١- جمع‌بندی

تجزیه‌ی وابستگی راهی برای تجزیه‌ی نحوی زبان طبیعی است که به صورت خودکار به تجزیه و تحلیل ساختار وابستگی جملات پرداخته و برای هر جمله‌ی ورودی یک گراف وابستگی ایجاد می‌کند. برچسب‌زنی اجزای سخن برای انجام تجزیه‌ی وابستگی یک پیش‌نیاز است. عموما تجزیه‌گرهای وابستگی پایپلاین برچسب‌زنی و تجزیه‌ی وابستگی را به صورت دو گام متوالی انجام می‌دهند. و بر روی متن‌های برچسب‌خورده که با آن‌ها آموزش دیده‌اند دارای دقت خوبی هستند. اما برای متنی از حوزه‌ی دیگر و یا یک متن خام جدید که تعداد واژه‌های جدید زیادی دارد دقت تجزیه‌ی وابستگی افت زیادی می‌کند. در همین راستا مدل‌های توأم معرفی شده‌اند تا بتوانند دو مسئله‌ی برچسب‌زنی اجزای سخن و تجزیه‌ی وابستگی را به صورت همزمان انجام دهند تا هم از اطلاعات نحوی مفید در حین برچسب‌زنی استفاده شود و هم بهبود در دقت برچسب‌زنی تأثیر مثبتی بر روی تجزیه‌ی وابستگی داشته باشد. در این راستا مدل‌های توأم به سه دسته‌ی مبتنی بر گذار، مبتنی بر گراف و روش‌های پشته‌ای تقسیم می‌شوند که روش‌های پشته‌ای دو تجزیه‌گر مبتنی بر گذار و مبتنی بر گراف را با هم ترکیب می‌کنند.

در این کار ابتدا بررسی مختصری روی برچسب‌زنی اجزای سخن و تجزیه‌ی وابستگی صورت گرفت سپس در مورد ضرورت استفاده از مدل‌های توأم برچسب‌زنی اجزای سخن و تجزیه‌ی وابستگی بحث شد. و در فصل ۳ به بررسی تعدادی از مدل‌های توأم ارائه شده پرداختیم.

٤-٢- کارهای آینده

از جمله کاربردهای تجزیه‌ی وابستگی در سیستم‌های پرسش و پاسخ[۵۷] و سیستم‌های ترجمه‌ی ماشینی است. در سیستم‌های پرسش و پاسخ نیاز به تجزیه‌ی جمله‌ی سوال و تمامی جملات پاسخ ممکن است[۳۲]. در تمامی این کاربردها نیاز به کار با متن خامی است که تجزیه‌گر قبلا روی آن آموزش ندیده است. بنابراین چون کلمات جدیدی دارد دقت تجزیه به میزان زیادی افت می‌کند. همچنین دریافتیم که مدل‌های پایپلاین از دو مشکل رنج می‌برند.۱- انتشار خطای ناشی از برچسب‌زنی در تجزیه‌ی وابستگی ۲- عدم استفاده از اطلاعات مفید نحوی در حین برچسب‌زنی. مدل‌های توأم چون دو مسئله‌ی برچسب‌زنی و تجزیه‌ی وابستگی را در یک زمان حل می‌کنند از این دو مشکل رنج نمی‌برند. مدل‌های

[57] Question Answering

توأم ارائه شده در سمینار سعی کرده‌اند هم دقت برچسب‌زنی و هم دقت تجزیه را افزایش دهند. اکثر مدل‌ها بر روی زبان چینی مورد بررسی قرار گرفته‌اند. تمام مدل‌ها دقت تجزیه را در حد خوبی بالا برده‌اند[۱, ۲, ۱۰, ۱۱]. و با بالابردن وزن ویژگی‌های اجزای سخن توانسته‌اند دقت برچسب‌زنی را نیز به میزان خوبی افزایش دهند[۳].

کارهای متعددی بر روی مدل‌های توأم انجام گرفته است. عمده‌ی کارها به زبان چینی است و تعداد کمی از کارها بر روی سه زبان دیگر انگلیسی، چکی و آلمانی مورد ارزیابی قرار گرفته است. تعدادی از کارها نیز کدشان در اختیار نیست و تعدادی نیز که کدشان موجود است عمدتا وابسته به زبان چینی است. بر همین اساس پروژه‌ای تحت عنوان **« ارائه و بهینه سازی مدل توأم برای برچسب زنی اجزای سخن و تجزیه وابستگی در متون خام زبان فارسی »** پیشنهاد می‌شود که در طی آن بایستی ابزار موجود مدل توأم موجود را برای زبان فارسی تطبیق داد و آن را راه‌اندازی کرد. سپس در ادامه مدل برچسب‌گذاری و نیز خصوصیات این مدل برای زبان فارسی بهینه‌سازی می‌شود. روش‌های تجزیه‌ی وابستگی به دو دسته‌ی مبتنی بر گذار و مبتنی بر گراف دسته‌بندی می‌شوند. اکثر کاربردهای تجزیه نیازمند اجرای بلادرنگ هستند، بنابراین در این پروژه به دلیل اجرای خطی روش‌های مبتنی بر گذار، تمرکز کار بر روی مدل توأم مبتنی بر گذار است.

مراجع

مراجع

1. Li, Z., et al. *Joint models for Chinese POS tagging and dependency parsing*. in *Proceedings of the Conference on Empirical Methods in Natural Language Processing*. 2011. Association for Computational Linguistics.
2. Hatori, J., et al. *Incremental Joint POS Tagging and Dependency Parsing in Chinese*. in *IJCNLP*. 2011.
3. Li, Z., et al. *A Separately Passive-Aggressive Training Algorithm for Joint POS Tagging and Dependency Parsing*. in *COLING*. 2012.
4. خلاش, م., بررسی روش‌های تجزیه در دستور وابستگی. ۱۳۹۰, دانشگاه علم و صنعت ایران.
5. سلطانی, م., سیستم برچسب گذاری و ابهام زدایی خودکار اجزای کلام برای پیکره متنی زبان فارسی. ۱۳۸۷, دانشگاه علم و صنعت ایران.
6. دانشگاه‌علم‌وصنعت‌ایران, فاز اول طرح جامع پیکره زبان فارسی با موضوع فاز اول مطالعاتی ایجاد پیکره متنی زبان فارسی in ,پیک‌متن‌فارس - ۲ - ث, ۱,۰ :1388 .شورای عالی اطلاع‌رسانی.
7. Ratnaparkhi, A. *A maximum entropy model for part-of-speech tagging*. in *Proceedings of the conference on empirical methods in natural language processing*. 1996.
8. Lafferty, J., A. McCallum, and F.C. Pereira, *Conditional random fields: Probabilistic models for segmenting and labeling sequence data*. 2001.
9. Collins, M. *Discriminative training methods for hidden markov models: Theory and experiments with perceptron algorithms*. in *Proceedings of the ACL-02 conference on Empirical methods in natural language processing-Volume 10*. 2002. Association for Computational Linguistics.
10. Bohnet, B. and J. Nivre. *A transition-based system for joint part-of-speech tagging and labeled non-projective dependency parsing*. in *Proceedings of the 2012 Joint Conference on Empirical Methods in Natural Language Processing and Computational Natural Language Learning*. 2012. Association for Computational Linguistics.
11. Liu, M.Z.W.C.T. and Z. Li, *Stacking Heterogeneous Joint Models of Chinese POS Tagging and Dependency Parsing.*
12. Nivre, J. *Non-projective dependency parsing in expected linear time*. in *Proceedings of the Joint Conference of the 47th Annual Meeting of the ACL and the 4th International Joint Conference on Natural Language*

Processing of the AFNLP: Volume 1-Volume 1. 2009. Association for Computational Linguistics.

13. Eisner, J.M. *Three new probabilistic models for dependency parsing: An exploration*. in *Proceedings of the 16th conference on Computational linguistics-Volume 1*. 1996. Association for Computational Linguistics.
14. McDonald, R., K. Crammer, and F. Pereira. *Online large-margin training of dependency parsers*. in *Proceedings of the 43rd Annual Meeting on Association for Computational Linguistics*. 2005. Association for Computational Linguistics.
15. McDonald, R.T. and F.C. Pereira. *Online Learning of Approximate Dependency Parsing Algorithms*. in *EACL*. 2006.
16. Carreras, X. *Experiments with a Higher-Order Projective Dependency Parser*. in *EMNLP-CoNLL*. 2007.
17. Koo, T. and M. Collins. *Efficient third-order dependency parsers*. in *Proceedings of the 48th Annual Meeting of the Association for Computational Linguistics*. 2010. Association for Computational Linguistics.
18. Zhang, Y. and J. Nivre. *Transition-based Dependency Parsing with Rich Non-local Features*. in *ACL (Short Papers)*. 2011.
19. Huang, L. and K. Sagae. *Dynamic programming for linear-time incremental parsing*. in *Proceedings of the 48th Annual Meeting of the Association for Computational Linguistics*. 2010. Association for Computational Linguistics.
20. Eisner, J., *Bilexical grammars and their cubic-time parsing algorithms*, in *Advances in Probabilistic and Other Parsing Technologies*. 2000, Springer. p. 29-61.
21. Bohnet, B. *Very high accuracy and fast dependency parsing is not a contradiction*. in *Proceedings of the 23rd International Conference on Computational Linguistics*. 2010. Association for Computational Linguistics.
22. Zhang, Y. and S. Clark. *Joint Word Segmentation and POS Tagging Using a Single Perceptron*. in *ACL*. 2008.
23. Crammer, K., et al., *Online passive-aggressive algorithms.* The Journal of Machine Learning Research, 2006. **7**: p. 551-585.
24. Crammer, K. and Y. Singer, *Ultraconservative online algorithms for multiclass problems.* The Journal of Machine Learning Research, 2003. **3**: p. 951-991.
25. Rush, A.M., *A tutorial on dual decomposition and Lagrangian relaxation for inference in natural language processing.* 2012.

26. Nivre, J. and R.T. McDonald. *Integrating Graph-Based and Transition-Based Dependency Parsers*. in *ACL*. 2008.
27. Charniak, E. and M. Johnson. *Coarse-to-fine n-best parsing and MaxEnt discriminative reranking*. in *Proceedings of the 43rd Annual Meeting on Association for Computational Linguistics*. 2005. Association for Computational Linguistics.
28. Petrov, S., et al. *Learning accurate, compact, and interpretable tree annotation*. in *Proceedings of the 21st International Conference on Computational Linguistics and the 44th annual meeting of the Association for Computational Linguistics*. 2006. Association for Computational Linguistics.
29. Petrov, S. and D. Klein. *Improved Inference for Unlexicalized Parsing*. in *HLT-NAACL*. 2007.
30. Yamada, H. and Y. Matsumoto. *Statistical dependency analysis with support vector machines*. in *Proceedings of IWPT*. 2003.
31. McDonald, R., *Discriminative learning and spanning tree algorithms for dependency parsing*. 2006, University of Pennsylvania.
32. Bouma, G., et al., *Question answering for Dutch using dependency relations*, in *Accessing Multilingual Information Repositories*. 2006, Springer. p. 370-379.