مدل سازی آماری زبان فارسی

محمد مهدی حاجی

بخش مهندسی و علوم کامپیوتر
دانشکده مهندسی، دانشگاه شیراز

بهار ۱۳۸۴

چکیده

این گزارش قسمتی از پژوهشهای انجام شده در زمینه طرح تحقیقاتی "مدلسازی آماری زبان فارسی" را ارایه می دهد. در این گزارش پس از ارایه مقدمه و تاریخچه به توصیف چگونگی طراحی و تنظیم مدلهای آماری برای کلمات و جملات فارسی پرداخته شده است. در هر مساله مدلسازی در ابتدا دو عامل اصلی می باید ابداع و تدوین شوند- ساختار مناسب و جامع برای مدل و سپس چگونگی تنظیم و محاسبه پارامترهای آن. در این تحقیق بر اساس مطالعات و بررسی های همه جانبه و با توجه به سوابق و کاربردهای متعدد، مدل پنهان مارکوف برای کلمات و مدلهای N-gram برای جملات در نظر گرفته شده اند. برای تنظیم و محاسبه پارامترهای مدل روشهای یادگیری هوشمند در ارائه خواهند شد. از آنجا که هر روش یادگیری نیاز به مجموعه ای از داده های آموزشی دارد لذا روشی کارآمد و جامع برای نمایش متن باید در نظر گرفته شود. در این تحقیق از استاندارد های بین المللی جهت ذخیره و پردازش فایلهای متن که شرح آن در این گزارش آمده است استفاده شده است. سپس به چگونگی به هنجار سازی و ایجاد یک لغتنامه جامع پرداخته شده است. برای داده های آموزشی طیف وسیعی از متون زبان فارسی در زمینه های مختلف ادبیات فارسی در نظر گرفته شده است طوری که مدل یاد گرفته شده مبتنی بر توزیع های مختلف و وسیعی از داده های آموزشی باشد.

بدیهی است که در گزارشهای بعدی چگونگی پیاده سازی و آموزش مدل کلمات و جملات و نهایتا نرم افزاری که بتواند در کاربردهای مدلسازی آماری زبان مورد استفاده قرار گیرد ارائه خواهد شد.

قسمت اول

مقدمه

یک مدل آماری زبان، احتمال جملات را توصیف میکند. یعنی با استفاده از آن میتوان با توجه به اطلاعات جاری، احتمال کلمات بعد را پیش بینی کرد. هرچند که مدل های دیگری از جمله مدلهای توانی (Exponential Models) (Rosenfeld et al., 2001) یا گرامرهای مستقل از متن (Context Free Grammars) (Corazza et al., 1993) برای مدل سازی زبانهای طبیعی پیشنهاد شده اند، اما در عمل مدل های آماری N-gram (Rosenfeld, 2000) از سایر روشها عملی تر و موثرتر بوده اند. زیرا علاوه بر سادگی پیاده سازی، نسبت به نویز هم مقاوم هستند.

کاربردهای عمده مدل آماری زبان در سیستم های اتوماتیک تشخیص گفتار (Automatic Speech Recognition) (Katz, 1987) و تشخیص متون ماشینی (OCR) (Lu et al., 1999) یا دستنویس (Handwritten Recognition) (Vinciarelli et al., 2004)

است. با توجه به حجم روزافزون اطلاعات و اهمیت پردازش خودکار گفتار و اسناد توسط کامپیوتر، میتوان به اهمیت مدلسازی زبان پی برد. از کاربردهای دیگر مدل زبان میتوان به این موارد اشاره کرد: استفاده در سیستم های ترجمه متون، پردازش زبانهای طبیعی (NLP) (Manning and Schütze, 1999)، پردازش و تصحیح خودکار لغات در پردازنده های متن، تبدیل متن به صدا (TTS) (Werner et al., 2004) و بهبود نتایج بازیابی اطلاعات متنی در موتورهای جستجو. در همه این کاربردها، عملکرد سیستم با استفاده از مدل آماری زبان بهتر میشود اما در سیستم های بازشناسی گفتار وجود مدل زبان حیاتی تر است؛ در واقع یک سیستم تشخیص گفتار بدون استفاده از مدلهای زبان به هیچ وجه نمی تواند دقت قابل قبولی داشته باشد.

با توجه به کاربردهای گسترده مدلهای زبان، در سالهای گذشته تحقیقات بسیاری برای مدل سازی زبانهای پراستفاده جهان و بیش از همه زبان انگلیسی انجام شده است و این تحقیقات همچنان ادامه دارند. جالب توجه است که مدلهای آماری زبان انگلیسی تقریبا از سال ۱۹۸۰ در سیستم های واقعی به کار رفته اند (Rosenfeld, 2000) اما برای مدل سازی زبان فارسی متاسفانه هنوز هیچ تحقیق گسترده و قابل اعتنایی صورت نگرفته است. البته با توجه به اینکه در کشور ما روند اتوماسیون عمر زیادی ندارد و عمده کاربردهای مدل آماری زبان در سیستمهای اتوماتیک پردازش و تشخیص گفتار و متن است، کمبود تحقیق در این زمینه کاملا قابل انتظار است. هدف اصلی این تحقیق و اولین گزارش آن که در پی می آید، جبران این کاستی و پایه ریزی اصول اولیه است. امید است که این پژوهش انگیزه و راهگشای کارهای بزرگتر بعدی باشد.

محتوای گزارش

در این گزارش پس از مقدمه و تاریخچه به بررسی چگونگی تهیه و تدارک داده های آموزشی برای یادگیری و تنظیم پارامترهای مدل مورد نظر پرداخته شده است. همانگونه که در شرح آمده است داده های آموزشی از طیف وسیعی از متون فارسی در زمینه ها و موضوعات مختلف جمع آوری شده اند. سپس روش به هنجارسازی این انبوه داده ها و روش استاندارد بین المللی یونی کد برای نمایش متون مورد بحث قرار گرفته است. در قسمت بعدی چگونگی ایجاد یک لغتنامه جامع و کامل از دیدگاه آماری پرداخته شده است. مدلهای آماری مورد نظر در این تحقیق، مبتنی بر مدلهای پنهان مارکوف برای کلمات و مدلهای N-gram برای جملات میباشند که پارامترهای آنها بر اساس الگوریتم های یادگیری که شرح داده خواهد شد محاسبه و تنظیم خواهند شد. در این گزارش مدل آماری مارکوف به طور کلی و مدل پنهان مارکوف به خصوص معرفی و شرح داده شده است.

داده های آموزشی

همانند هر مدل آماری دیگر در این سیستم نیز به داده هایی برای آموزش یـا در واقـع تنظـیم پارامترهای آزاد مدل احتیاج داریم. برای تهیه یک مدل آماری، داده هـای آموزشـی بایـد دارای توزیع احتمالی باشند که بعدا مدل با آنها سروکار دارد؛ یعنی به طـور کلـی داده هـای آموزشـی باید فضای احتمال را پوشش دهند. یعنی اگر بخواهیم مدل آماری به دست آمـده در همـه جـا قابل استفاده باشد، واضح است که باید انواع مختلف متون - از جمله ادبی، سیاسـی، اقتصـادی، طنز، عقیدتی و ... - را برای آموزش سیستم به کار گیریم.

تهیه داده های آموزشی

برای تهیه داده های آموزشی با چنین حجم زیادی تنها راه عملی اسـتفاده از متـون موجـود در شبکه اینترنت است. داده های آموزشی که در این تحقیق به کـار گرفتـه شـده انـد، شـامل ۱) متون عمومی: مجلات سروش (که ۹۵٪ حجم کل داده های آموزشی را به خـود احتصـاص داده اند) و بعضی از مقالات موجود در سایت الشیعه؛ ۲) متون تاریخی: از جملـه کلیلـه و دمنـه و ۳) اشعار: از جمله دیوان حافظ و برخی از دفاتر شعر نیما یوشیج و فروغ فرخ زاد. حجم کل نوشـته جات آموزشی که در نهایت به دست آوردیم، در حدود ۱۰۰ مگابایت (با فرمت یونی کـد) اسـت شامل حدود ۷۳۰۰۰۰۰ کلمه می باشد. لازم به توضیح است که مجلات سروش شامل نشـریات سروش کودکان، سروش نوجوان، سروش جوان، سـروش بـانوان، سـروش اندیشـه و هفتـه نامـه سروش میباشد که این نشریات شامل انواع مختلف متون فارسی هستند. در زیـر از هـر یـک از این نشریات بخشهایی آورده شده است:

سروش کودکان:

خروس به روی بام ویرانه‌ای پرید و قوقولی قو سر داد. روباه از فرصت استفاده کرد. و به سر خروس پرید و خروس را به دندان گرفت و فرار کرد...

این طوری بود که کورینلیوس تصمیم گرفت کروکودیل ها را بـه حـال خودشـان بگذارد و از کنار رودخانه دور شود امـا هنـوز راه زیـادی نرفتـه بـود کـه بـه یـک میمون برخورد کرد...

سروش نوجوان:

اتفاق‌هایی هم که در کارتون‌ها می‌افتد، خیلی‌وقت‌ها اتفاق‌های واقعی نیسـتند یـا اگـر اتفاق‌های واقعی‌اند، نتایجی غیرمعمول دارند؛ مثلاً گربه‌ای پایش به چیزی گیر می‌کنـد و به‌جای این که زمین بخورد، بدنش کش می‌آید...

سروش جوان:
زیبایی آمریکایی بیشترین جوایز اسکارامسال را برد. چیزی که از قبل میشد حدس زد در میان برندگان جوایز، نام *کوین اسپیسی* دیده می‌شود...
گزارشی که باعث می شود همان روز عصر تمام آن مزرعه و کشتارگاه قرنطینه و تمام سی *صدوچهل* خوک و سی ودو گاو موجود در آن ها معدوم شوند...

سروش بانوان:
بعد مردم می‌گویند چرا زندگی‌های الان این قدر بی دوام است؟ چرا بین زن و شوهرها جنگ و دعواست؟ چرا فلان است، چرا بهمان است؟...

سروش اندیشه:
مکتب فلسفهِ جاودانه (حکمت *خالده*) با دیدگاهی همبسته است که مطابق آن دانشی آغازین، یا دانشی قدسی وجود دارد که بر متافیزیک عام این حقیقت نمایی مبتنی است...

هفته نامه سروش:
سفر قندهار، فیلمی درباره مردم افغانستان است...
هرچه ز بیگانه وخیل تواند جمله در این خانه طفیل تواند...
رضا عطاران را حتما میشناسید، بازیگر مجموعه های طنز...

همانطور که در این نمونه ها مشاهده میشود، مجموعه متون آموزشی شامل اسم های خاص ایرانی (رضا عطاران) و غیر ایرانی (کوین اسپیسی)، عدد (صدوچهل) و ترکیبات تخصصی (حکمت خالده) نیز هست. بنابراین کاملا قابل انتظار است که لغتنامه فارسی که از این متون استخراج خواهد شد، لغات بسیار زیادی داشته باشد که بسیاری از آنها ممکن است در لغتنامه های عادی وجود نداشته باشند.

نرمال سازی فایلهای متنی

فایلهایی متنی که از اینترنت جمع آوری کرده ایم، به فرمت HTML هستند. در این فایلها، کاراکترهای فارسی با سیستمهای کدگذاری متفاوتی (Cp1256، Unicode و UTF8) نشان داده میشوند. در اولین مرحله نرمال سازی، فایل ورودی را با هر سیستم کدگذاری به فایلی یونی کد تبدیل میکنیم. برای این کار یک برنامه جاوا نوشته شده است. در زیر ابتدا توضیحاتی

درباره استاندارد یونی کد ارائه میکنیم و سپس نرمال سازی های بیشتری را که روی متون انجام میشود شرح خواهیم داد.

استاندارد یونی کد

استاندارد یونی کد، استاندارد جهانی کدگذاری نویسه هاست که برای ارائه متون برای پردازش کامپیوتري به‌کار میرود. این استاندارد با ویرایش دوم استاندارد بین‌المللی SO/IEC 10646-1:2000 کاملا سازگار است و همان نویسه‌ها و کدهای ISO/IEC 10646 را دارد. استاندارد یونی‌کد اطلاعات بیشتری نیز در مورد نویسه‌ها و کاربردهایشان فراهم کرده است، پس در واقع هر پیاده‌سازی سازگار با یونی کد، با ISO/IEC 10646 نیز سازگار است و روش رسمی پیاده سازی استاندارد ISO/IEC 10646 یونی کد است.

استاندارد یونی کد را پیشتازان صنعت کامپیوتر، شرکت هایی چون اوراکل، آی بی ام، اپل، سان، سای بیس، مایکروسافت، هیولت پاکارد، و بسیاری دیگر پذیرفته اند. استانداردهایی چون XML، جاوا، اکماسکریپت (جاواسکریپت)، LDAP، CORBA 3.0، WML، و غیره، یونی کد را ملزم می دانند. یونی کد در بسیاری از سیستم عامل ها، همه ی مرورگرهای اخیر، و بسیاری از محصولات دیگر پشتیبانی می شود. یونی کد مستقل از محیط، مستقل از برنامه، و مستقل از زبان به همه نویسه ها اعداد یکتایی اختصاص میدهد. تا قبل از اختراع یونی کد، صدها سیستم کدگذاری مختلف برای تخصیص این اعداد وجود داشت. نویسه های هیچ کدگذاری ای به تنهایی کافی نبود: مثلا اتحادیه اروپا به چندین کدگذاری مختلف برای پوشاندن همه ی زبان هایش نیاز داشت. حتی برای زبانی مثل انگلیسی نیز هیچ کدگذاری ای به تنهایی برای همه حروف، علائم نقطه گذاری، و نمادهای فنی متداول کافی نبود. این سیستم ها با هم تعارض نیز داشتند. یعنی، دو کدگذاری ممکن بود از اعداد یکسان برای دو نویسه مختلف، یا از اعداد مختلف برای نویسه های یکسان استفاده کنند. کامپیوترها، خصوصا سرورهای شبکه، از کدگذاری های مختلف و متعددی پشتیبانی می کردند و بنابراین هرگاه داده ها از کدگذاری ها یا محیط های مختلف عبور می کردند، در معرض خطر تحریف قرار می گرفتند.

یونی کد امکان کدگذاری همه نویسه‌های مورد استفاده در نوشتن زبان‌های دنیا را فراهم می سازد. این استاندارد از یک کدگذاری ۱۶بیتی استفاده می کند که برای بیش از ۶۵۰۰۰ نویسه جا فراهم می کند. اگر چه ۶۵۰۰۰ نویسه برای کدگذاری اکثر نویسه‌هایی که در زبان‌های مهم دنیا استفاده می شود کافی است، با این وجود، یونی کد و ISO 10646 شیوه گسترشی به نام UTF-16 فراهم کرده‌اند که امکان اضافه کردن حدود یک میلیون نویسه دیگر را نیز فراهم

میکند. این دامنه برای کلیه نویسه‌های عالم، از جمله پوشش کامل همه خط‌های باستانی نیز کافی است.

یونی کد برای کلیه های نویسه‌های مورد استفاده در زبانهای عمده دنیا کد تعیین کرده است. به‌علت فراخ بودن فضای تخصیص نویسه، این استاندارد بسیاری از نمادهای لازم برای حروفچینی با کیفیت بالا را نیز در بر گرفته است. از خطهای مورد پشتیبانی این استاندارد میتوان به لاتین (در بر گیرنده اکثر زبان‌های اروپایی)، سیریلیک (روسی، صربی)، یونانی، عربی (شامل عربی، فارسی، اردو، کردی)، عبری، هندی، ارمنی، آسوری، چینی، کاتاکانا و هیراگانا (ژاپنی)، و هانگول (کره ای) اشاره کرد. به‌علاوه، تعداد زیادی نماد ریاضی، علائم نقطه گذاری و علامت های متفرقه در این استاندارد وجود دارد. این استاندارد برای علامت های ترکیب شونده یا اعراب ها نیز کدهایی در نظر گرفته است که از جمله آنها علامت هایی چون «˜» هستند که در ترکیب با حروف پایه، حروف تغییر لحن یافته ای چون «ñ» را می سازند. آخرین نسخه یونی کد، در مجموع، ۴۹۱۹۴ نویسه دارد. به‌علاوه، ۶۴۰۰ کد نیز برای مصرف خصوصی در نظر گرفته شده است که برنامه‌نویسان میتوانند از آنها برای نویسه ها و نمادهای خودشان استفاده کنند.

به طور کلی، اصول یونی کد به این شرح می باشند: نویسه‌های شانزده‌بیتی، ترتیب مفهومی (در مقابل دیداری)، کارایی، یکی سازی (اختصاص یک کد به نویسه‌های مشترک در چند زبان مختلف)، نویسه نه شکل (یک «ع»، و نه چهارتا: «ع»، «ع»، «ع»، «ع»)، ترکیب پویا، بار معنایی (حرف بودن، مقدار عددی و ...)، دنباله های معادل (امکان ذخیره‌سازی یک متن به دو شکل مختلف ولی معادل)، متن ساده (و نه مفاهیمی مثل تغییر قلم، جدول‌بندی، و صفحه آرایی)، قابلیت تبدیل (هر متن موجود در قالب یک مجموعه‌نویسه سنتی باید بدون از بین رفتن معنا قابل تبدیل به یونی کد باشد).

یونی کد شیوه ای نیز برای کدگذاری ۸ بیتی متون مشخص کرده است که نویسه ها را پس از اعمال یک تابع خاص به کدشان، به صورت دنباله هایی که از یک تا چهار بایت دارند نگه می دارد. این شیوه که با نام UTF-8 شناخته می شود، به این خاطر که نویسه های ASCII ر ا عینا حفظ میکند و در نتیجه، هم برنامه های قدیمی راحت تر با آن کنار می آیند و هم طول پرونده های لاتین را زیاد نمی کند، بسیار محبوب است. در واقع بسیاری از سیستم هایی که ادعای پشتیبانی یونی کد را می کنند، تنها UTF-8 را پشتیبانی می کنند و پرونده های یونی کدی، اعم از کاربردهای اینترنتی یا موارد استفاده محلی، عمدتا در قالب UTF-8 ذخیره شده اند. در استاندارد یونی کد، نویسه های فارسی در بلوک مربوط به خط عربی قرار دارند. این بلوک برای دربرگرفتن نویسه های زبانهایی که از خط عربی استفاده می کنند، مثل فارسی، اردو، پشتو، سندی، و کردی گسترش یافته است. این بلوک نشانه های قرآنی از قبیل نشانه های سجده و پایان آیه، و علائم وقف را نیز در بر دارد.

در یونی کد با وجود یکی سازی کدهای حروف مشترک، برای حروف فارسی ای که بار معنایی یا نمایشی متفاوت با حروف عربی دارند، نویسه های جداگانه در نظر گرفته شده است. یعنی کلیه حروف خاص فارسی (پ، چ، ژ، گ) و نیز «ک» و «ی» ی فارسی که با حرف مشابه در عربی تفاوت نمایشی دارند، مکان جداگانه ای به خود اختصاص داده اند. کلیه اعراب های متداول حضور دارند و میان شکل فارسی/ اردو و عربی ارقام نیز به علت شکل و رفتار متفاوت تفاوت هایی منظور گشته است.

از طرف دیگر، علائم نقطه گذاری ای چون نقطه و فاصله که شکل یکسانی در خط های لاتین و عربي دارند، کد یکسان دارند. علائمی چون پرانتز نیز، بسته به جهت متن، آینه ای میشوند، یعني به طور مثال، نویسه ۰۰۲۸ نماینده «پر انتز باز» است، و نه «پر انتز سمت چپ». یونی کد اتصال مجازی و فاصله مجازی را نیز تحت نامهای «اتصال با عرض صفر» و «بی اتصالی با عرض صفر» به رسمیت می شناسد. به خاطر سازگاری با استاندارد های موجود در بعضی از کشورهای عربی، ISO 10646 و نتیجتا یونی کد بلوک جداگانه ای را نیز به شکل های مختلف حروف عربی اختصاص داده است که استفاده از آنها شدیدا منع شده است. این بلوک معمولا فقط برای تعیین جای شکل های مختلف حروف در قلمها به کار میرود.

همچنین این استاندارد توضیحات مفصل و دقیقی درباره شیوه های پیاده سازی، از جمله شیوه «متصل سازی حروف» و «نمایش متون راست به چپ و دوجهته» دارد که برنامه نویس را از مراجعه به راهنمای محلی بی نیاز می سازد. خواننده میتواند برای توضیحات بیشتر در این زمینه به مستندات یونی کد مراجعه کند (www.unicode.org).

نرمال سازی های بیشتر

با توجه به مطالبی که درباره استاندارد یونی کد گفتیم، میتوان دریافت که گاهی ممکن است کلمات یکسان با مجموعه ای از نویسه های غیر یکسان نمایش داده شوند. به عنوان مثال کلمه «یک» میتواند چهار شکل نمایش مختلف داشته باشد، که از ترکیب دو «ی» (فارسی و عربی) و دو «ک» (فارسی و عربی) به دست می آیند. بنابراین، در اولین مرحله نرمال سازی، هر حرفی را که در استاندارد یونی کد با دو یا چند نویسه مختلف نمایش داده میشود، فقط با یکی از این نویسه ها نشان می دهیم؛ به عنوان مثال هرجا «ک» عربی باشد، به جای آن «ک» فارسی میگذاریم. علاوه بر این، حرف «ـ» (تطویل) هم که برای کنترل طول کلمات استفاده میشود و هیچ اطلاعات مفیدی را دربر ندارد برای پردازش های بعدی حذف میکنیم.

مرحله بعدی، حذف اِعراب کلمات است، با وجود اینکه گاهی تفاوت دو کلمه فقط با اِعراب مشخص میشود، اما چون در اکثریت نوشته جات آموزشی که از اینترنت جمع آوری شده اند، کلمات اِعراب گذاری نشده اند، مجبور هستیم اِعراب کلمات را به طور کلی حذف کنیم. به بیان دیگر، کلمات یا باید همواره دارای اِعراب باشند، یا همواره دارای اِعراب نباشند و چون کلمات

موجود در نوشته جات آموزشی همواره دارای اِعراب نیستند، پس تنها راه برای نرمال سازی این است که کلمات هیچ اِعرابی نداشته باشند.

باید توجه داشت که قواعد نرمال سازی که در بالا ذکر کردیم، اگرچه بسیاری از موارد را پوشش می دهند اما هنوز حالتهایی وجود دارند که کلمات یکسان با مجموعه ای از نویسه های غیر یکسان نمایش داده میشوند که عمدتا به علت غیراستاندارد بودن شیوه نگارش است. به عنوان مثال، بعضی از نویسنده ها ترجیح می دهند «ء» آخر کلمات را ننویسند و به این ترتیب مثلا «اشیاء» و «اشیا» از نظر سیستم ما، که یک مجموعه از حروف فارسی پشت سر هم را به عنوان یک کلمه در نظر میگیرد، دو کلمه متفاوتند. به طور مشابه، در بعضی از نوشته جات آموزشی، به جای «ة» آخر کلمات از «ه» استفاده شده است و به این ترتیب مثلا «حمیدة» و «حمیده» دو کلمه متفاوت میشوند. اما مشکل بزرگتر، سر هم یا جدا نوشتن کلماتی است که از بخشهای مختلفی درست شده اند که این مشکل عمدتا در نگارش افعال استمراری فارسی و جمع بستن کلمات با «ها» پیش می آید. به عنوان مثال، با توجه به تعریف کلمه از نظر سیستم ما، «می روم» شامل دو کلمه «می» و «روم» است، اما «میروم» یک کلمه متفاوت است. و به همین ترتیب، «درخت ها» شامل دو کلمه «درخت» و «ها» است اما «درختها» یک کلمه متفاوت است.

نکته قابل ذکر دیگر این است که سیستم ارائه شده چون رویکردی کاملا آماری دارد، تلاشی برای ریشه یابی لغات انجام نمیدهد و بنابراین به عنوان مثال «کتابم»، «کتابت»، «کتابش» و ... از نظر این سیستم، کلمات متفاوتی هستند. این مساله ناشی از اتصال ضمایر ملکی به انتهای کلمات است.

استخراج لغتنامه فارسی

پس از نرمال سازی مجموعه نوشته جات (داده های) آموزشی، می توان لغتنامه فارسی را استخراج کرد. بار دیگر یاد آوری می کنیم که در این مدل، یک دنباله پشت سر هم از نویسه ها را به عنوان یک کلمه در نظر می گیریم. همانطور که قابل انتظار است، مجموعه نوشته جات آموزشی شامل نویز هم می باشد، مثلا وقتی که یک کلمه اشتباه تایپ شده باشد. گاهی اوقات هم بسته به نوع متن، کلمات محاوره ای در متن وجود دارند؛ مثلا «بتونیم» به جای «بتوانیم». و همانطور که قبلا ذکر شد، کلماتِ خاص زبانهای خارجی و اعداد هم کلماتی از مجموعه نوشته جات هستند. برای کاهش دادن این کلمات ناخواسته، یک روال ساده را در پیش می گیریم: فرکانس تکرار تمامی کلمات موجود در مجموعه نوشته جات را محاسبه میکنیم و سپس لغاتی را که فرکانس تکراری کمتر از یک حد آستانه (از پیش تعیین شده) دارند حذف میکنیم. در این تحقیق مقدار آستانه را ۳ قرار دادیم، یعنی در واقع کلماتی را که کمتر از ۳ بار پیش آمده بودند، به عنوان نویز در نظر گرفته و حذف کردیم. با این رویه، یک لغتنامه شامل

حدود ۴۱۰۰۰ کلمه به دست آمد. این مجموعه بر حسب فرکانس تکرار مرتب شده است، تا اگر برای کاربردی خاص مثلا به هزار لغت پر فرکانس تر (پر کاربردتر) فارسی احتیاج است بتوان به راحتی از هزار لغت ابتدای این مجموعه استفاده کرد. ده عضو ابتدای این لغتنامه به ترتیب اینها هستند: و، می، به، در، که، از، را، این، است، با. همانطور که قابل انتظار است پر فرکانس ترین لغت فارسی حرف ربط «و» است. بدیهی است که قرار گرفتن کلمه «می» به عنوان دومین کلمه لغتنامه به دلیل وجود تعداد تکرارهای کلمه «مِی» نیست بلکه به دلیل تعداد تکرار زیاد «می» در بخش ابتدایی افعال استمراری (مانند می روم) است که به شکل غیر چسبان نوشته (تایپ) شده است. حتی اگر در متون اولیه تعدادی کلمه «مِی» هم وجود داشته باشد، چون در مرحله نرمال سازی، اعراب کلمات حذف می شوند، اینها عملا با «می» فرقی نخواهند داشت و به افزایش تعداد تکرارهای کلمه «می» کمک خواهند کرد. نکته جالب توجه این است که پرفرکانس ترین لغات فارسی (یعنی اعضای نخست لغتنامه) حروف ربط هستند و به طور مشابه در زبان انگلیسی پرفرکانس ترین کلمه The است و کلمات پرفرکانس بعدی حروف ربط انگلیسی از جمله of، and، to هستند.

در زیر بعضی از کلمات لغتنامه ۴۱۰۰۰ کلمه ای را مشاهده می کنید:

شانگهای	کورتکس	نوزدهم	اینتراکتیو
صدوچهل	بودیسم	جدیدترین	آفساید
سیدمحمدرضا	انتگرال	یورونیوز	نوددرصد
پدرومادرها	ارتودنسی	پرتابل	بلندبلند
ازتهران	توتیا	پراهمیت	دوقسمتی
گیگز	مغیلان	اسکیزوفرنی	اوپک
کامیونیکیشن	تغار	شیشلول	مستثناست
بتونیم	هزارتو	امپایر	مستظرفه

کاملا واضح است که علیرغم تلاشی که برای حذف لغات نویز کردیم، هنوز لغات ناخواسته ای در لغتنامه وجود دارند. البته قابل انتظار است که با افزایش حجم نوشته جات آموزشی وبالا بردن حد آستانه ی حداقلِ فرکانسِ تکرار، کلمات ناخواسته کاهش یابند. اما نکته مثبت این لغتنامه، داشتن لغاتی است که در لغتنامه های عادی وجود ندارند و حتی ممکن است لغات فارسی یا عربی هم نباشند، مثلا انتگرال و اینترنت، اما به دلیل استفاده زیاد واقعا بخشی از زبان فارسی جدید شده باشند. اشکالات تایپی رایج و حتی بی معنی بودن برخی اعضای لغتنامه استخراج شده اهمیت چندانی ندارد؛ زیرا مدل آماری بعدا با متونی شامل همین لغات سروکار خواهد داشت.

مدلهای مارکوف

چون مدلی که برای کلمات ارائه می دهیم، بر مبنـای مـدلهای مـارکوف اسـت، در ایـن بخـش مقدمه ای بر مدلهای مارکوف و سپس مدلهای پنهان مارکوف ارائه می دهیم.

خروجی یک فرآیند در جهان واقعی به شکل یک سیگنال پیوسته یا گسسته مشاهده می شـود. یک مساله حیاتی در علوم ساختن مدلهایی برای این سیگنالهای واقعی است. مدل سـازی یـک سیگنال مزایای فراوانی به همراه دارد. اولا، مدل، پایه ای برای توصیف تئـوری سـیگنال فـراهم میکند که میتواند برای پردازش سیگنال استفاده شود تا خروجی خواص مطلوبی داشـته باشـد. ثانیا، مدل میتواند اطلاعات بسیار مفیدی درباره منبع سیگنال بدهد، بدون اینکـه احتیـاجی بـه خود منبع باشد. نهایتا و از همه مهمتر، مدلها میتوانند در عمل بـه خـوبی کـار کننـد و امکـان تحقق سیستم های عملی مهمی را فراهم می آورند.

بسته به نوع و خواص سیگنال، راه های مختلفی برای مدل کردن یک سـیگنال وجـود دارد. بـه طور کلی، یک سیگنال میتواند معین یا نامعین (تصـادفی یـا آمـاری) باشـد. مـدلهای معـین از بعضی خواص شناخته شده سیگنال استفاده می کنند و مقادیر پارامترهای مدل را تخمین مـی زنند. در طرف دیگر، در مدلهای آماری، یک فرآیند تصادفی، سیگنال را توصـیف میکنـد. بـرای کاربرهایی نظیر تشخیص گفتار یا دستخط که با نویز و عدم قطعیـت همـراه هسـتند، مـدلهای آماری از کارایی بهتری برخوردارند. مدلهای پنهان مارکوف (Rabiner, 1989)، که همچنـین منابع مارکوف یا توابع آماری زنجیره های مارکوف نامیده می شوند، در تئوری مخابرات یکـی از پرکاربرترین مدلهای آماری هستند.

دسته مهمی از فرآیند های تصادفی، فرآیندهای مـارکوف اسـت کـه دارای خواصـی اسـت کـه مطالعه ریاضی آنها را امکان پذیر میکند. در جهان واقعی، معمولا مطلوب است کـه یـک دنبالـه متغیرهای تصادفی وابسته را - که مقدار هر متغیر به عنصر (یا عناصر) قبلی در دنبالـه بسـتگی دارد - بررسی کنیم. در یک فرآیند مارکوف مقدار متغیر تصادفی جاری برای پیش بینی مقـدار متغیرهای تصادفی آینده کافی است. به بیان دیگر، وقتی عنصر جاری را داشـته باشـیم، عناصـر آینده از عناصر گذشته مستقل شرطی انـد. فـرض کنیـد $X = (X_1, \ldots, X_T)$ دنبالـه متغیرهـای تصادفی باشد که مقادیری از فضای محدود $S = \{s_1, \ldots, s_N\}$ میگیرند. خـواص مـارکوف بـا دو رابطه زیر بیان میشوند:

(۱) $$P(X_{t+1} = s_k \mid X_1, X_2, \ldots, X_t) = P(X_{t+1} = s_k \mid X_t)$$

(۲) $$P(X_{t+1} = s_k \mid X_t) = P(X_2 = s_k \mid X_1)$$

خاصیت دوم را تغییر ناپذیری با زمان می نامیم. اگر دنباله X هر دو خاصیت مـارکوف را داشـته باشد، آن را یک زنجیره مارکوف می نامیم.

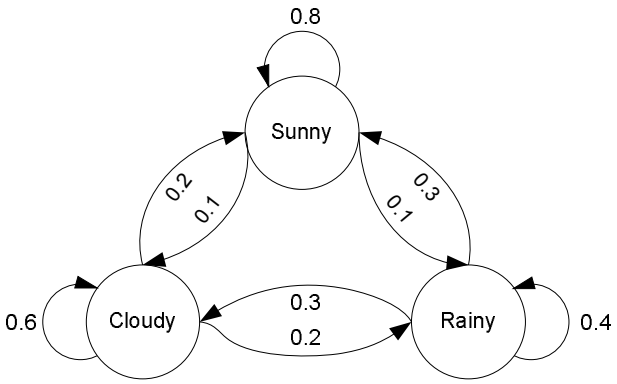
یک زنجیره مارکوف را می توان با بردار تصادفی وضعیت اولیه $\prod$ و ماتریس تصـادفی انتقـال A به طور کامل توصیف کرد:

$$\pi_i = P(X_1 = s_i) \quad (۳)$$

$$a_{ij} = P(X_{t+1} = s_j \mid X_t = s_i) \quad (۴)$$

که $\pi_i \geq 0, \forall i$ و $\sum_{i=1}^{N} \pi_i = 1$ و $a_{ij} \geq 0, \forall i, j$ و $\sum_{j=1}^{N} a_{ij} = 1, \forall i$.

برای روشن کردن این مطالب، مثالی درباره پیش بینی وضع هوا را در نظر بگیرید کـه در آن بـا استفاده از تاریخچه مشاهدات وضع هوا در گذشته می خواهیم هوای فردا را حدس بزنیم. بـرای سادگی فرض می کنیم هوا سه حالت بیشتر نداشته باشد: خورشیدی، ابری و بـارانی و وضـعیت هوا در طول یک روز یکسان است؛ یعنی تغییر حالتی در وسط روز اتفاق نمی افتـد. اگـر فـرض کنیم زنجیره وضعیت هوا در روزهای متوالی مارکوف است (کـه البتـه در جهـان واقعـی فـرض درستی نیست)، آنگاه ماشـین حالـت محـدود شـکل ۱ بـا احتمـالات انتقـال (تغییـر وضـعیت) دلخواهی که روی پیکان ها داده شده است این زنجیره مارکوف را نشان می دهـد. توجـه کنیـد که مجموع احتمالات پیکانهای خروجی در هر یک از سه حالت ۱ است. از شکل ۱ واضـح اسـت که مدل مارکوف را می توان به عنوان یک ماشین حالت محدود نامعین (تصادفی) به شمار آورد که در آن احتمالات روی پیکانهای تغییر وضعیت قرار گرفته اند.



شکل ۱ - مدل مارکوف مثال پیش بینی وضع هوا

فرض کنیـد $s_1 = Sunny$ ، $s_2 = Cloudy$ و $s_3 = Rainy$ باشـد و در اولـین روز هـوا خورشـیدی باشد. آنگاه:

$$\prod = (1.0, 0.0, 0.0)$$

$$A = \begin{bmatrix} 0.8 & 0.1 & 0.1 \\ 0.2 & 0.6 & 0.2 \\ 0.3 & 0.3 & 0.4 \end{bmatrix}$$

محاسبه احتمال یک دنباله از وضعیتها $X_1, \ldots, X_K$ برای یک زنجیره مارکوف به سادگی با رابطه زیر انجام میشود:

$$\begin{aligned} P(\mathrm{X}_1, \ldots, \mathrm{X}_K) &= P(X_1)\, P(X_2 \mid X_1)\, P(X_3 \mid X_1, X_2) \ldots P(X_K \mid X_1, \ldots, X_{K\text{-}1}) \\ &= P(X_1)\, P(X_2 \mid X_1)\, P(X_3 \mid X_2) \ldots P(X_K \mid X_{K\text{-}1}) \\ &= \pi_{X_1} \prod_{t=1}^{K-1} a_{X_t X_{t+1}} \end{aligned} \qquad (۵)$$

بنابراین در مثال بالا احتمال اینکه هوا در هفت روز آینده به ترتیب خورشیدی، خورشیدی، بارانی، بارانی خورشیدی، ابری و خورشیدی باشد یا در واقع احتمال $O = s_1, s_1, s_3, s_3, s_1, s_2, s_1$ میتواند به شکل زیر محاسبه شود:

$$\begin{aligned} P(O \mid \mathrm{Model}) &= \pi_{s1} P(s_1 \mid s_1)\, P(s_1 \mid s_1)\, P(s_3 \mid s_1)\, P(s_3 \mid s_3)\, P(s_1 \mid s_3)\, P(s_2 \mid s_1)\, P(s_1 \mid s_2) \\ &= \pi_1 a_{11}\, a_{11}\, a_{13}\, a_{33}\, a_{31}\, a_{12}\, a_{21} \\ &= 1.0\ (0.8)\ (0.8)\ (0.1)\ (0.4)\ (0.3)\ (0.1)\ (0.2) \\ &= 1.536 \times 10^{-4} \end{aligned} \qquad (۶)$$

به طور کلی وقتی از مدلهای مارکوف صحبت می کنیم، منظور ما مدلهای مارکوف مرتبه اول است که در آن تاریخچه ای به طول ۱ (یعنی یک عنصر قبلی) برای پیش بینی رفتار آینده استفاده میشود. اما گاهی اوقات برای پیش بینی حالتهای آینده تاریخچه بزرگتری لازم است. در یک مدل مارکوف مرتبه n، برای پیش بینی حالت بعدی، از n حالت قبلی استفاده می شود. اما همواره می توان با تغییر شکل نمایش فضای حالت، هر مدل مارکوف مرتبه n را به یک مدل مارکوف مرتبه یک تبدیل کرد. بنابراین از نظر تئوری، فرض مارکوف مرتبه اول، محدود کننده نیست.

قسمت دوم

مدلهای پنهان مارکوف

مدلهای پنهان مارکوف (Hidden Markov Models)، که به اختصار HMM نامیده میشوند، یکی از قوی ترین ابزارها برای پردازش سیگنالها میباشند. انواع مختلف HMM علیرغم محدودیتهایی که دارند، هنوز پراستفاده ترین تکنیک در سیستمهای مدرن بازشناسی گفتار و تشخیص متون هستند. مدل پنهان مارکوف، کل الگوی ورودی را به عنوان یک بردار

ویژگی تکی مدل نمیکند، بلکه رابطه بین بخشهای متوالی یک الگو را استخراج میکند، زیرا هر بخش نسبت به کل ورودی کوچکتر و بنابراین مدل سازی آن ساده تر است.

یک HMM را در واقع میتوان یک ماشین حالت محدود (Finite State Machine) احتمالاتی به حساب آورد که هر حالت با یک تابع تصادفی مرتبط است. فرض میشود که در یک دوره گسسته از زمان t، مدل در یک حالت است و با یک تابع تصادفی از آن حالت خروجی ای (مشاهده ای) تولید میکند. بر مبنای تابع احتمال انتقال حالت جاری، مدل مارکوف در زمان $t+1$ تغییر حالت میدهد. دنباله حالتهایی که مدل از آن میگذرد معمولا پنهان است، و تنها یک تابع احتمالاتی از آن آشکار است، که مشاهدات تولید شده به وسیله تابع تصادفی مربوط به حالتها است، به همین دلیل در نام گذاری این مدلها از صفت "پنهان" استفاده شده است. یک HMM را در واقع میتوان یک فرآیند تصادفی به طور ناتمام (جزئی) مشاهده شده در نظر آورد. یک HMM با عناصر زیر توصیف میشود:

(۷) N: تعداد حالتهای مدل

(۸) $S = \{s_1, s_2, ..., s_N\}$: مجموعه حالتها

(۹) $\prod = \{\pi_i = P(s_i \text{ at } t = 1)\}$: احتمالات حالت اولیه

(۱۰) $A = \{a_{ij} = P(s_j \text{ at } t+1 \mid s_i \text{ at } t)\}$: احتمالات تغییر حالت

(۱۱) M: تعداد علائم قابل مشاهده (تولید شده)

(۱۲) $V = \{v_1, v_2, ..., v_M\}$: مجموعه علائم قابل مشاهده

(۱۳) $B = \{b_i(v_k) = P(v_k \text{ at } t \mid s_i \text{ at } t\}$: احتمالات تولید (انتشار) علائم قابل مشاهده

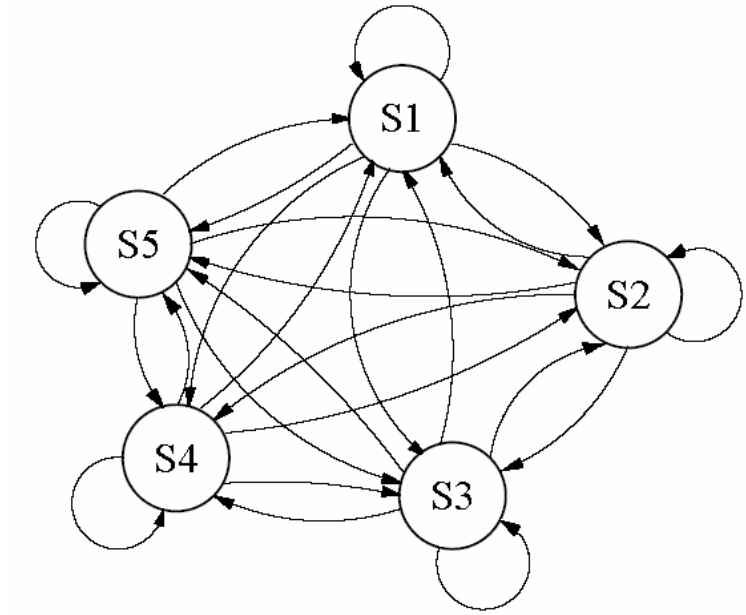
(۱۴) O_t: علامت مشاهده شده در زمان t

(۱۵) T: طول دنباله مشاهدات

(۱۶) $\lambda = (A, B, \prod)$: نماد خلاصه برای مدل پنهان مارکوف

واضح است که بر احتمالات $\prod$، A و B سه قید وجود دارد: $\sum_{i=1}^{N}\pi_i = 1$، $\sum_{j=1}^{N}a_{ij} = 1,\ \forall i$ و $\sum_{k=1}^{M}b_i(v_k) = 1,\ \forall i$.

ساختار ماتریس احتمالات تغییر حالت A توپولوژی HMM را تعیین میکند. اگر $a_{ij} \neq 0\ \ \forall i, j$ یعنی هر حالت مدل از هر حالت دیگر با یک گذر قابل رسیدن باشد، مدل "کاملا متصل" یا "ارگودیک" (Ergodic) نامیده میشود (شکل ۲). برای مدل سازی کلمات زبان که بعدا شرح داده خواهد شد، از مدل ارگودیک استفاده خواهد شد.



شکل ۲- یک مدل مارکوف ارگودیک پنج حالته

یک توپولوژی دیگر که در کاربردهای بازشناسی گفتار و متن بسیار پراستفاده است، توپولوژی "چپ به راست" (LR) یا "بکیس" (Bakis) میباشد که در آن حالتهای با شماره پایین تر، مشاهدات نخستین را تولید میکنند. ترتیب زمانی در HMM های چپ به راست با قرار دادن صفرهای ساختاری در مدل به شکل قیدهای $\prod = \{1, 0, ..., 0\}$ و $a_{ij} = 0,\ i > \mathrm{j}$ اعمال میشود، که یعنی مدل از اولین حالت (حالت با کمترین شماره و در واقع سمت چپ ترین حالت) شروع میکند و در هر حالت تغییر فقط میتواند به حالتهای با شماره بالاتر صورت گیرد. به عنوان یک محدودیت دیگر، بیشتر اوقات در HMM های چپ به راست، اندازه پرشهای رو به جلو در هر حالت محدود میشود و به این ترتیب از تغییر حالت زیاد جلوگیری میشود، یعنی برای یک Δ ثابت، $a_{ij} = 0,\ j > i + \Delta$ در نظر گرفته میشود.

مثال زیر به درک کاربرد مدلهای پنهان مارکوف کمک میکند. فرض کنید یک نفر برای مدتی در اتاقی زندانی است و میخواهد از وضعیت هوای بیرون اطلاع داشته باشد. تنها اطلاعاتی که او از دنیای بیرون دارد این است که آیا کسی که هر روز برای او وعده غذایش را می آورد با چتر وارد میشود یا نه؛ بنابراین برای مشاهده حمل چتر $V = \{True, False\}$. برای سادگی فرض میشود که هوا تنها سه وضعیت دارد: آفتابی، ابری و بارانی، و یک روز معادل یک بازه زمانی است، یعنی وضعیت هوا در طول روز تغییر نمیکند. فرض کنید احتمال حمل چتر در یک روز به شرط اینکه هوا آفتابی باشد 0.1 باشد، همچنین احتمال حمل چتر به شرط ابری بودن 0.3 و احتمال حمل چتر به شرط بارانی بودن 0.7 باشد. هدف این است که فرد از مشاهدات (حمل کردن یا نکردن چتر) درباره وضعیت هوای بیرون (که از او پنهان است) نتیجه ای بگیرد. فرض

کنید w_i وضعیت هوا در روز i باشد، و متغیر بولی u_i به این معنی باشد که در آن روز چتر مشاهده میشود یا خیر. با استفاده از قانون بیس (Bayes):

$$P(w_1,...,w_n \mid u_1,..u_n) = \frac{P(u_1,..u_n \mid w_1,...,w_n)P(w_1,...w_n)}{P(u_1,...,u_n)} \quad (۱۷)$$

احتمال $P(w_1, ...,w_n)$ معادل مدل مارکوف مثال قبل است، و $P(u_1, ...,u_n)$ احتمال از پیش معلوم دیدن دنباله ای خاص از رخدادهای حمل کردن یا نکردن چتر است. اگر فرض شود که به ازای همه i ها، به شرط w_i، u_i از هر u_j و w_j به ازای هر $j \neq i$ مستقل باشد، آنگاه احتمال $P(u_1,...,u_n \mid w_1,...,w_n)$ این گونه محاسبه میشود $\prod_{i=1}^{n} P(u_i \mid w_i)$.

در مثال پیش بینی وضع هوا (و خیلی از مسائل دیگر) میتوان از احتمال پیشین $P(u_1, ...,u_n)$ صرفنظر کرد زیرا از وضع هوا مستقل است. بر مبنای فرض مارکوف مرتبه اول، معیار محتمل بودن (Likelihood)، که با احتمال متناسب است، اینگونه محاسبه میشود:

$$P(w_1,..., w_n \mid u_1, ..., u_n) \propto$$

$$L(w_1,..., w_n \mid u_1, ..., u_n) = P(u_1,...,u_n \mid w_1,..., w_n)\, P(w_1, ..., w_n)$$

$$= \prod_{i=1}^{n} P(u_i \mid w_i) \prod_{i=1}^{n} P(w_i \mid w_{i-1}) \quad (۱۸)$$

فرض کنید روز حبس شدن فرد آفتابی بوده باشد، و روز بعد مسوول غذا با چتر وارد شده باشد. مطلوب است پیش بینی وضعیت هوای روز بعد:

در ابتدا با این فرض که روز بعد آفتابی بوده باشد معیار محتمل بودن را حساب میکنیم:

$$L(w_2 = Sunny \mid w_1 = Sunny, u_2 = True) = P(u_2 = True \mid w_2 = Sunny)\ .$$

$$P(w_2 = Sunny \mid w_1 = Sunny) = 0.1\ (0.8) = 0.08 \quad (۱۹)$$

سپس با این فرض که روز بعد ابری بوده باشد معیار محتمل بودن را حساب میکنیم:

$$L(w_2 = Cloudy \mid w_1 = Sunny, u_2 = True) = P(u_2 = True \mid w_2 = Cloudy)\ .$$

$$P(w_2 = Cloudy \mid w_1 = Sunny) = 0.3\ (0.1) = 0.03 \quad (۲۰)$$

و سرانجام با این فرض که روز بعد بارانی بوده باشد معیار محتمل بودن را حساب میکنیم:

$$L(w_2 = Rainy \mid w_1 = Sunny, u_2 = True) = P(u_2 = True \mid w_2 = Rainy)\ .$$

$$P(w_2 = Rainy \mid w_1 = Sunny) = 0.7\ (0.1) = 0.07 \quad (۲۱)$$

بنابراین پر احتمال ترین حالت این است که روز بعد آفتابی بوده باشد.

سه مساله اساسی مدلهای پنهان مارکوف

در کاربردهای HMM، نیازمند حل حداقل یکی از سه مساله زیر هستیم:

مساله ۱. اگر مدل $\lambda = (A, B, \prod)$ را داشته باشیم، چطور میتوانیم احتمال $P(O \mid \lambda)$ یعنی احتمال وقوع دنباله مشاهده $O = O_1, O_2, ..., O_T$ به شرط مدل را به شکل موثری حساب کنیم؟

مساله ۲. اگر مدل λ و بردار مشاهده O را داشته باشیم، چطور دنباله حالت $S = s_1, s_2, ..., s_T$ را انتخاب کنیم که $P(O, S \mid \lambda)$ یعنی احتمال مشترک دنباله مشاهده $O = O_1, O_2, ..., O_T$ و دنباله حالت S به شرط مدل، ماکزیمم شود؟ به بیان دیگر، هدف این است که دنباله حالتی پیدا کنیم که مشاهدات را به بهترین نحو توصیف کند.

مساله ۳. اگر بردار مشاهده O را داشته باشیم، چطور پارامترهای مدل $\lambda = (A, B, \prod)$ را تنظیم کنیم که احتمال $P(O \mid \lambda)$ یا $P(O, S \mid \lambda)$ ماکزیمم شود. به بیان دیگر، هدف پیدا کردن (یا در واقع آموزش) مدلی است که داده های مشاهده شده را به بهترین نحو توصیف کند.

حل مساله ۱

در این مساله هدف محاسبه احتمالی است که مدل λ دنباله مشاهده O را تولید کند. ساده ترین روش برای محاسبه $P(O \mid \lambda)$ پیدا کردن $P(O \mid S, \lambda)$ برای یک دنباله حالت ثابت S و ضرب آن در $P(S \mid \lambda)$ و جمع برای تمام دنباله های حالت ممکن به طول T است:

$$P(O \mid \lambda) = \sum_{S} P(O \mid S, \lambda).P(S \mid \lambda) \qquad (۲۲)$$

چون $P(S \mid \lambda) = \pi_{s_1} a_{s_1 s_2} a_{s_2 s_3} ... a_{s_{T-1} s_T}$ و $P(O \mid S, \lambda) = b_{s_1}(O_1) b_{s_2}(O_2) .. b_{s_T}(O_T)$ معادله (۲۲) میتواند اینگونه نوشته شود:

$$P(O \mid \lambda) = \sum_{S} \pi_{s_1} b_{s_1}(O_1) a_{s_1 s_2} b_{s_2}(O_2) ... a_{s_{T-1} s_T} b_{s_T}(O_T) \qquad (۲۳)$$

محاسبه احتمال با استفاده از رابطه (۲۳) عملی نیست زیرا N^T دنباله حالت وجود دارد و بنابراین $(2T\text{-}1)N^T$ عمل ضرب و N^T-1 عمل جمع لازم است. بنابراین استفاده از یک روش عملی و کاراضروری است. خوشبختانه دو روش برای این کار وجود دارد: الگوریتم رو به جلو (Forward) و الگوریتم رو به عقب (Backward) که شرح آنها در زیر می آید:

الگوریتم رو به جلو برای هر حالت s متغیر رو به جلو $\alpha_t(s)$ را که به شکل زیر تعریف میشود محاسبه میکند:

$$\alpha_t(s) = P(O_1, O_2, ..., O_t, s_t = s \mid \lambda) \qquad (۲۴)$$

که احتمال دیدن جزئی دنباله مشاهده تا زمان t و بودن در حالت s به شرط مدل λ است. روال سه مرحله ای زیر $\alpha_t(s)$ را برای تمام حالات و زمانها حساب میکند:

۱- مقدار دهی نخستین:

$$\alpha_1(s) = \pi_s b_s(O_1),\ 1 \le s \le N \quad (۲۵)$$

۲- استقرا:

$$\alpha_{t+1}(r) = [\sum_{s=1}^{N} \alpha_t(s) a_{sr}] b_r(O_{t+1}),\ 1 \le r \le N, 1 \le t \le T-1 \quad (۲۶)$$

این احتمال رو به جلوی قرار گرفتن در حالت r در زمان t+1 بر مبنای احتمال مشترک متغیرهای رو به جلوی همه حالتها در زمان t و احتمالات انتقال از هر یک از آنها به حالت r است. به این دلیل میتوان احتمال را به این شکل محاسبه کرد که از هر یک از N حالت در زمان t میتوان به طور مستقل (در زمان t+1) به حالت r (با احتمال a_{sr}) رسید.

۳- خاتمه:

$$P(O \mid \lambda) = \sum_{s=1}^{N} \alpha_T(s) \quad (۲۷)$$

محاسبه متغیرهای رو به جلو برای تمام حالتها و در تمام زمانها به $N(N\text{-}1)(T\text{-}1)+(N\text{-}1)$ عمل جمع و $N + N(N+1)(T\text{-}1)$ عمل ضرب احتیاج دارد؛ که از مرتبه N^2T است، در مقایسه با مرتبه محاسبات در روش مستقیم که TN^T بود.

الگوریتم رو به عقب کاملا شبیه به الگوریتم رو به جلو است با این تفاوت که در اینجا متغیر رو به عقب $\beta_t(s)$ که در زیر تعریف میشود برای هر حالت در جهت عکس محاسبه میشود:

$$\beta_t(s) = P(O_{t+1}, O_{t+2}, \ldots, O_T \mid s_t = s, \lambda) \quad (۲۸)$$

این احتمال دنباله مشاهده از t+1 تا T به شرط مدل λ و قرار گرفتن در حالت s در زمان t است. شبیه $\alpha_t(s)$، $\beta_t(s)$ هم میتواند برای تمام حالات و زمانها با یک الگوریتم سه مرحله ای محاسبه شود:

۱- مقدار دهی نخستین:

$$\beta_T(s) = 1,\ 1 \le s \le N \quad (۲۹)$$

۲- استقرا:

(۳۰) $$\beta_t(s) = \sum_{r=1}^{N} a_{sr} b_r(O_{t+1}) \beta_{t+1}(r),\ 1 \le s \le N, t = T-1, T-2, ..., 1$$

۳- خاتمه:

(۳۱) $$P(O \mid \lambda) = \sum_{s=1}^{N} \pi_s b_s(O_1) \beta_1(s)$$

محاسبه $P(O \mid \lambda)$ با استفاده از متغیرهای رو به عقب نیز شامل محاسباتی از مرتبه N^2T است.

حل مساله ۲

در این مساله میخواهیم پراحتمال ترین دنباله حالت (یعنی قسمت پنهان مدل) متناظر با یک دنباله مشاهده را پیدا کنیم. الگوریتم معروف "ویتربی" (Viterbi, 1967) یک الگوریتم برنامه نویسی پویا (Dynamic Programming) برای پیدا کردن این دنباله (مسیر) بهینه است. این الگوریتم بهترین دنباله حالت را در هر لحظه از زمان برای هر یک از N حالت نگه میدارد، و در نهایت بهترین مسیر را برای هر یک از N حالت به عنوان حالتی که آخرین مشاهده در آن صورت گرفته است محاسبه میکند، و از بین این مسیرها، مسیر دارای بیشترین احتمال را به عنوان مسیر بهینه کلی انتخاب می کند.

الگوریتم چهار مرحله ای ویتربی همان استراتژی الگوریتم رو به جلو را دنبال میکند، اما با این تفاوت که عمل جمع با ماکزیمم (یا مینیمم) جایگزین میشود. انتخاب ماکزیمم یا مینیمم بستگی به این دارد که معیار بهینگی چه انتخاب شده باشد (بدیهی است که اگر معیار انتخاب شده احتمال باشد، باید ماکزیمم شود و اگر هزینه باشد باید مینیمم شود). برای دنباله مشاهده $O = O_1, O_2, ..., O_T$ و مدل λ الگوریتم به شکل زیر بیان میشود:

۱- مقدار دهی نخستین:

(۳۲) $$\delta_1(s) = \pi_s b_s(O_1)$$

(۳۳) $$\psi_1(s) = 0,\ 1 \le s \le N$$

که $\delta_t(s)$ نشان دهنده وزنهای انباشته شده است وقتی که مدل در زمان t در حالت s قرار دارد، و $\psi_t(s)$ نمایانگر حالتی است که در زمان t-1 پایین ترین هزینه (بالاترین احتمال) مربوط به انتقال حالت به s در زمان t را داشته است.

۲- استقرا:

(۳۴) $$\delta_t(s) = \max_{1 \le r \le N} [\delta_{t-1}(r) a_{rs}] b_s(O_t)$$

$$\psi_t(s) = \arg\max_{1\le r\le N}[\delta_{t-1}(r)a_{rs}],\ 1\le s\le N, 2\le t\le T \quad (۳۵)$$

۳- خاتمه:

$$P^* = \max_{1\le s\le N}[\delta_T(s)] \quad (۳۶)$$

$$q_T^{\ *} = \arg\max_{1\le s\le N}[\delta_T(s)] \quad (۳۷)$$

۴- برگشت به عقب برای پیدا کردن مسیر بهینه:

$$q_t^{\ *} = \psi_{t+1}(q_{t+1}^{\ *}),\ t = T-1, T-2, ..., 1 \quad (۳۸)$$

و حالا $Q^* = \{q_1^*, q_2^*, ..., q_T^*\}$ دنباله حالت بهینه است و P^* احتمال مشترک دنباله مشاهده O و دنباله حالت بهینه Q^* است.

شبیه به الگوریتم های رو به جلو و رو به عقب، پیچیدگی محاسباتی الگوریتم ویتربی از مرتبه N^2T میباشد.

البته توجه به این نکته ضروری است که پیاده سازی مستقیم الگوریتم بالا میتواند باعث به وجود آمدن مشکل پاریز (underflow) شود، زیرا احتمالاتی که حساب میشوند شامل ضرب اعداد کوچک است که به سرعت باعث میشود محدوده اعداد تولید شده از محدوده پیش بینی شده اعداد ممیز شناور کامپیوتر خارج شود. برای حل این مشکل، الگوریتم ویتربی طوری تغییر داده میشود تا به جای احتمال با لگاریتم احتمال کار کند. این تکنیک نه تنها مشکل پاریز را حل میکند بلکه روند محاسبات را نیز سرعت میبخشد، زیرا عمل جمع خیلی سریعتر از ضرب است. باید توجه کرد که الگوریتم ویتربی یک الگوریتم زمان اجراست نه یک الگوریتم آموزش که معمولا به شکل آفلاین انجام میشود، بنابراین پیاده سازی سریع این الگوریتم بسیار مطلوب و ضروری است. نسخه کارا و عملی الگوریتم ویتربی در زیر می آید:

۰- پیش پردازش

$$\tilde{\pi}_s = \log(\pi_s),\ 1\le s\le N \quad (۳۹)$$

$$\tilde{a}_{rs} = \log(a_{rs}),\ 1\le r,s\le N \quad (۴۰)$$

$$\tilde{b}_s(O_t) = \log(b_s(O_t)),\ 1\le s\le N, 1\le t\le T \quad (۴۱)$$

۱- مقدار دهی نخستین:

$$\tilde{\delta}_1(s) = \tilde{\pi}_s + \tilde{b}_s(O_1) \quad (۴۲)$$

$$\psi_1(s) = 0,\ 1\le s\le N \quad (۴۳)$$

۲- استقرا:

$$\widetilde{\delta}_t(s) = \max_{1\le r\le N}[\widetilde{\delta}_{t-1}(r) + \widetilde{a}_{rs}] + \widetilde{b}_s(O_t) \quad (۴۴)$$

$$\psi_t(s) = \arg\max_{1\le r\le N}[\widetilde{\delta}_{t-1}(r) + \widetilde{a}_{rs}],\ 1 \le s \le N, 2 \le t \le T \quad (۴۵)$$

۳- خاتمه:

$$P^* = \max_{1\le s\le N}[\widetilde{\delta}_T(s)] \quad (۴۶)$$

$$q_T{}^* = \arg\max_{1\le s\le N}[\widetilde{\delta}_T(s)] \quad (۴۷)$$

۴- برگشت به عقب برای پیدا کردن مسیر بهینه:

$$q_t{}^* = \psi_{t+1}(q_{t+1}{}^*),\ t = T-1, T-2, \ldots, 1 \quad (۴۸)$$

و حالا $Q^* = \{q_1{}^*, q_2{}^*, \ldots, q_T{}^*\}$ دنباله حالت بهینه است و $\exp(P^*)$ احتمال مشترک دنباله مشاهده O و دنباله حالت بهینه Q^* است.

شاید متوجه شده باشید که الگوریتم ویتربی تنها شامل عملیات ضرب است که با لگاریتم گیری تبدیل این اعمال ضرب به سادگی تبدیل به جمع میشوند. اما الگوریتمهای رو به جلو و رو به عقب شامل عملیات جمع هستند. در اینجا اگر لازم باشد مشکل پاریز اعداد ممیز شناور میتواند با همان لگاریتم گیری حل شود، تنها لازم است $\log(x+y)$ حساب شود که با استفاده از تکنیک زیر قابل انجام است (Manning and Schütze, 1999):

```
if y - x > log big
    return y;
  else if x - y > log big
    return x;
  else
      return min(x,y) + log(exp(x - min(x,y)) + exp(y - min(x,y)));
```

که *big* یک ثابت به حد کافی بزرگ مثلا 10^{30} است.

حل مساله ۳

بر مبنای اینکه چه احتمالی برای ماکزیمم کردن انتخاب شود، دو روش کلی برای تخمین پارامترهای مدل (آموزش) وجود دارد: الگوریتم segmental k-means (Juang and Rabiner, 1990) که پارامترهای مدل را تنظیم میکند تا $P(O, Q^*|\lambda)$ ماکزیمم شود، که

Q^* دنباله حالت بهینه متناظر با دنباله مشاهده O است. و الگوریتم Baum-Welch (Rabiner, 1989) که پارامترهای مدل را تنظیم میکند تا $P(O \mid \lambda)$ به یک ماکزیمم برسد، در اینجا $P(O \mid \lambda)$ میتواند جمع احتمالات $P(O, S \mid \lambda)$ روی تمام دنباله حالتهای ممکن S تعبیر شود، یعنی این الگوریتم روی دنباله حالت خاصی متمرکز نمیشود. معمولا الگوریتم segmental k-means برای آموزش ترجیح داده میشود زیرا در مقایسه با الگوریتم Baum-Welch به محاسبات خیلی کمتری احتیاج دارد، و همچنین برای خیلی از کاربردهای مدلسازی و کدگشایی، معیار $P(O, Q^* \mid \lambda)$ کاملا طبیعی به نظر میرسد. در دو قسمت زیر شرح الگوریتم ها آورده میشود.

الگوریتم Segmental K-Means

مانند هر الگوریتم آموزش، الگوریتم segmental k-means نیز به تعدادی دنباله مشاهده (آموزشی) احتیاج دارد. فرض کنید به تعداد w از چنین دنباله هایی موجود باشد. هر دنباله $O = O_1, O_2, ..., O_{T_i}$ شامل T_i بردار مشاهده است، بنابراین در مجموع $\sum_{i=1}^{w} T_i$ بردار مشاهده وجود دارد. اگر تنها یک دنباله دراز موجود باشد، میتوان آن را به تعداد اختیاری دنباله های کوچک تقسیم کرد و از آنها برای آموزش استفاده کرد. فرض میشود هر علامتِ (Symbol) مشاهده یک بردار با اندازه (بُعد) یک یا بیشتر باشد و بدیهی است که بردارهای مشاهده باید دارای بُعد یکسان باشند. همچنین در مدلهای پنهان مارکوف گسسته که تا اینجا مورد بحث قرار گرفته اند لازم است که این علامتها متعلق به یک مجموعه محدود باشند. الگوریتم شامل مراحل زیر است:

۱- N بردار آموزشی $C_1, C_2, ..., C_N$ را به طور تصادفی انتخاب کنید و هر یک از بردارهای آموزشی باقیمانده را به یکی از این N بردار که کمترین فاصله (مثلا اقلیدسی) را تا آن دارد منسوب کنید. بنابراین N دسته (خوشه) شکل میگیرد که هر کدام یک حالت (با شماره ای بین ۱ تا N) نامیده میشود. نماد $O_t \in s$ یعنی t اُمین علامت از یک دنباله مشاهده به حالت (دسته) s منسوب شده است. انتخاب اولیه دسته ها بر HMM نهایی تاثیری نمیگذارد، اما میتواند تعداد تکرارهای لازم برای آموزش مدل را تعیین کند. برای اینکه انتخاب اولیه دسته ها تا حد امکان توزیع پراکنده ای داشته باشد، یک استراتژی خوب برای وقتی که $w \geq N$ این است که C_1 اولین بردار مشاهده اولین دنباله انتخاب شود، C_2 دومین بردار مشاهده دومین دنباله انتخاب شود و ... (Dugad and Desai, 1996). این مرحله مقدار دهی نخستین مناسبی برای کل روال آموزش آماده میکند.

۲- احتمالات نخستین و انتقال را به شکل زیر محاسبه کنید:

(۴۹) $$\hat{\pi}_s = \frac{\text{number of occurrences of } \{O_1 \in s\}}{\text{total number of occurrences of } O_1}, \ 1 \le s \le N$$

(۵۰) $$\hat{a}_{rs} = \frac{\text{number of occurrences of } \{O_t \in r \text{ and } O_{t+1} \in s\}}{\text{total number of occurrences of } \{O_t \in r\}}, \ 1 \le r,s \le N, \ 1 \le t \le T_i - 1$$

۳- ماتریس میانگین و کوواریانس را به شکل زیر محاسبه کنید:

(۵۱) $$\hat{\mu}_s = \frac{1}{N_s} \sum_{O_t \in s} O_t, \ 1 \le s \le N$$

(۵۲) $$\hat{V}_s = \frac{1}{N_s} \sum_{O_t \in s} (O_t - \hat{\mu}_s)^T (O_t - \hat{\mu}_s), \ 1 \le s \le N$$

۴- توزیع احتمال را برای هر بردار مشاهده و برای هر حالت محاسبه کنید:

(۵۳) $$\hat{b}_s(O_t) = \frac{1}{(2\pi)^{\frac{D}{2}} |\hat{V}_s|^{\frac{1}{2}}} \exp(-\frac{1}{2}(O_t - \hat{\mu}_s)\hat{V}_s(O_t - \hat{\mu}_s)^T)$$

ثابت شده است که برای دسته وسیعی از توابع چگالی احتمال از جمله گاوسی، الگوریتم همگرا میشود. در اینجا تابع گاوسی به طور اختیاری انتخاب شده است.

۵- با احتمالات جدید با استفاده از الگوریتم ویتربی برای هر دنباله آموزشی، دنباله حالت بهینه Q^* را پیدا کنید. اگر انتساب حالت قبلی یک بردار مشاهده متفاوت از حالت بهینه تخمین زده شده متناظر باشد آن را به حالت جدید منسوب کنید؛ یعنی اگر $q_t^* = s$ آنگاه O_t را به s منسوب کنید.

۶- اگر در مرحله ۵ هر کدام از بردارهای مشاهده به حالتی جدید (یعنی حالتی مخالف با حالت انتساب یافته قبلی اش) منسوب شد، آنگاه مراحل ۲ تا ۶ باید تکرار شوند و در غیر این صورت الگوریتم متوقف میشود.

الگوریتم Baum-Welch

این الگوریتم جزء دسته الگوریتم های موسوم به ماکزیمم کردن توقع (EM) میباشد. الگوریتم EM روش بسیار پراستفاده ای برای یادگیری در مسائلی است که در آنها متغیرهای مشاهده نشده (پنهان) وجود دارند. یک الگوریتم EM با تخمین زدن مکرر مقدار مورد انتظار (Expected Value) هر متغیر پنهان با استفاده از فرضهای جاری و سپس محاسبه مجدد متحمل ترین فرضها (maximum likelihood hypothesis) با استفاده از مقادیر مورد

انتظار متغیرها، محتمل ترین فرضیه ها را جستجو میکند. به بیان دیگر، در مرحله اول، فرضهای جاری برای تخمین مقدار متغیرهای پنهان استفاده میشوند و در مرحله دوم مقادیر این متغیرها برای بهبود دادن فرضیه ها استفاده میشود. میتوان اثبات کردکه چنین روندی نهایتا فرضیه هایی دارای بیشترین احتمال محلی را پیدا میکند. یعنی ممکن است الگوریتم EM همانند بسیاری از الگوریتمهای تکراری ممکن است نتواند جواب بهینه سراسری را پیدا کند، اما همواره یک بهینه محلی را پیدا میکند (Mitchell, 1997).

تخمین اولیه فرضها (پارامترها)، یعنی ساخت HMM اولیه، میتواند با هر روشی انجام شود، اما یک تخمین اولیه معقول میتواند با استفاده از چهار مرحله اول الگوریتم segmental k-means به دست آید.

قبل از ارائه فرمولهای اصلی باید چند مفهوم و نماد را معرفی کنیم. $\gamma_t(s) = P(s_t = s \mid O, \lambda)$ را در نظر بگیرید، که احتمال قرار داشتن در حالت s در زمان t به شرط مدل λ و دنباله مشاهده O است. با استفاده از قانون بیس داریم:

$$\gamma_s(t) = \frac{P(s_t = s, O \mid \lambda)}{P(O \mid \lambda)} = \frac{\alpha_t(s)\beta_t(s)}{P(O \mid \lambda)},\ 1 \le s \le N \qquad (۵۴)$$

که $\alpha_t(s)$ و $\beta_t(s)$ متغیرهای رو به جلو و رو به عقب هستند.

همچنین $\xi_t(r,s) = P(s_t = r, s_{t+1} = s \mid O, \lambda)$ را تعریف میکنیم که احتمال قرار داشتن در حالت r در زمان t و انتقال به حالت s در زمان $t+1$ به شرط مدل λ و دنباله مشاهده O است. با استفاده از قانون بیس و خاصیت کازال بودن زنجیره مارکوف (رابطه (۱)) میتوان نشان داد:

$$\xi_t(r,s) = \frac{\alpha_t(r)a_{rs}b_s(O_{t+1})\beta_{t+1}(s)}{P(O \mid \lambda)},\ 1 \le r, s \le N \qquad (۵۵)$$

اگر $\gamma_t(s)$ از $t = 1$ تا T جمع زده شود، تعداد دفعات قابل انتظاری که حالت s دیده شده است به دست می آید، و اگر تنها تا T-1 جمع زده شود، تعداد دفعات قابل انتظاری که از حالت s انتقالی انجام شده است به دست می آید. به طور مشابه، اگر $\xi_t(r,s)$ از $t = 1$ تا T جمع زده شود، تعداد دفعات قابل انتظاری که انتقالی از حالت r به حالت s انجام گرفته است به دست می آید:

$$\sum_{t=1}^{T-1} \gamma_t(s) = \text{expected number of transitions from state } s,\ 1 \le s \le N \qquad (۵۶)$$

$$\sum_{t=1}^{T-1} \xi_t(r,s) = \text{expected number of transitions from state } r \text{ to state } s,\ 1 \le r,s \le N \qquad (۵۷)$$

رابطه بین $\gamma_t(r)$ و $\xi_t(r,s)$ میتواند با جمع کردن $\xi_t(r,s)$ روی s به دست آید:

$$\gamma_t(r) = \sum_{s=1}^{N} \xi_t(r,s),\ 1 \le r \le N \quad (۵۸)$$

و حالا فرمولهای تخمین مجدد Baum-Welch به شکل زیر تعریف میشوند:

$$\hat{\pi}_s = \gamma_1(s),\ 1 \le s \le N \quad (۵۹)$$

$$\hat{a}_{rs} = \sum_{t=1}^{T-1} \xi_t(r,s) \Big/ \sum_{t=1}^{T-1} \gamma_t(r),\ 1 \le r,s \le N \quad (۶۰)$$

$$\hat{b}_s(v_k) = \sum_{t=1, O_t = v_k}^{T} \gamma_t(s) \Big/ \sum_{t=1}^{T} \gamma_t(s),\ 1 \le s \le N \quad (۶۱)$$

فرمول تخمین مجدد برای π_s به سادگی برابر است با احتمال قرار داشتن در حالت s در زمان ۱. فرمول تخمین مجدد برای a_{rs} برابر است با نسبت تعداد دفعات قابل انتظار انتقال حالت از r به s به تعداد دفعات قابل انتظاری که از r انتقال حالتی انجام شده است. و سرانجام، فرمول تخمین مجدد برای $b_s(v_k)$ برابر است با نسبت تعداد دفعات قابل انتظار قرار داشتن درحالت s و مشاهده علامت v_k به تعداد دفعات قابل انتظار قرار داشتن درحالت s .

مدل (پنهان) مارکوف برای کلمات فارسی

در استفاده از HMM برای مدلسازی، یک مساله اساسی تعیین تعداد حالتهای مدل است؛ در بیشتر اوقات، ارتباط از پیش معلومی بین تعداد حالتها و تعداد علائم قابل مشاهده وجود ندارد. مساله تعیین تعداد حالتها در HMM ها مشابه مساله تعیین تعداد لایه های مخفی و تعداد نرونهای آنها در شبکه های عصبی است. چون هنوز تئوری یا روشی اصولی برای انتخاب تعداد حالتهای HMM وجود ندارد، به عنوان یک راه حل ساده، معمولا تعداد حالتهای مدل را متناسب با تعداد علائم قابل مشاهده، که کاملا مشخص است، در نظر میگیرند. مثلا در کاربردهای تشخیص متون ماشینی، معمولا پنج (تا ده) حالت به هر حرف (کاراکتر) تخصیص داده میشود و بنابراین تعداد حالتهای مدل برابر میشود با پنج (تا ده) برابر اندازه مجموعه کاراکترها (علائم قابل مشاهده). اما در مدلی که برای کلمات فارسی پیشنهاد میدهیم، هر حالت مدل را با یک حرف الفبای فارسی متناظر میکنیم، بنابراین مدل قسمت پنهانی ندارد؛ اینگونه مدلها را "مدلهای آشکار مارکوف" (Visible Markov Models) یا به اختصار VMM مینامند. چون ترتیب زمانی در مدل کلمات اهمیتی ندارد و به طور طبیعی هر حالت مدل (یعنی هر حرف) باید با یک انتقال از هر حالت دیگر قابل رسیدن باشد، معقول است که برای این مدل توپولوژی کاملا متصل (ارگودیک) را برگزینیم.

در مدلهای VMM، چون دنباله حالت بهینه از پیش مشخص است، سه مساله مدلهای HMM به شکل خیلی ساده تری حل میشود. در مدل پیشنهادی برای کلمات، دنباله های مشاهده (آموزشی) همان کلمات لغتنامه استخراج شده هستند، هرچند که میتوان برای ساخت (آموزش) مدل، یا در واقع محاسبه احتمالات اولیه، انتقال و انتشار (مشاهده)، از الگوریتم های استاندارد Segmental K-Means یا Baum-Welch استفاده کرد، اما این احتمالات را به سادگی میتوان از روابط زیر به دست آورد:

$$\pi_i = \frac{\text{No. of words starting with letter } s_i}{\text{Lexicon size}} \qquad (۶۲)$$

$$a_{ij} = \frac{\text{No. of transitions from letter } s_i \text{ to } s_j}{\text{Total number of transitions from letter } s_i} \qquad (۶۳)$$

$$b_i(v_k) \approx \begin{cases} 1 & \text{if } s_i = v_k \\ 0 & \text{otherwise} \end{cases} \qquad (۶۴)$$

حالا با داشتن این احتمالات، برای محاسبه $P(O \mid \lambda)$، یعنی حل مساله ۱، بجای استفاده از الگوریتم رو به جلو یا رو به عقب میتوان به سادگی از رابطه زیر استفاده کرد:

$$P(O \mid \lambda) = \pi_{O_1} a_{O_1 O_2} \ldots a_{O_{T-1} O_T} \qquad (۶۵)$$

نتایج آزمایشی مدل ارائه شده کلمات فارسی

در زیر چند نمونه از احتمالات اولیه و انتقال که با استفاده از فرمولهای (۶۲) و (۶۳) محاسبه شده اند آورده شده است:

$\pi_{\text{ب}} = 3946 / 41000 \approx 0.096$
$\pi_{\text{ژ}} = 81 / 41000 \approx 0.002$
$\pi_{\text{م}} = 4590 / 41000 \approx 0.112$

$a_{\text{ب} \to \text{ا}} = 1795 / 8029 = 0.224$
$a_{\text{ب} \to \text{ب}} = 86 / 8029 = 0.011$
$a_{\text{ب} \to \text{ی}} = 1035 / 8029 = 0.129$
$a_{\text{ژ} \to \text{خ}} = 0.003$
$a_{\text{ف} \to \text{ف}} = 0.0005$
$a_{\text{ذ} \to \text{ظ}} = 0.0$

چون مدل پنهان (یا آشکار) مارکوف یک مدل مولد است، با استفاده از مدل ارائه شده برای کلمات میتوان کلماتی را نیز تولید کرد، که در واقع محتمل ترین دنباله های مشاهده از دید مدل هستند. در جدول۱ نمونه هایی از کلمات تولید شده توسط مدل را مشاهده میکنید:

سه حرفی	چهار حرفی	پنج حرفی	شش حرفی
بکس	**جاشو**	**حدیده**	**رزماند**
رزک	فسور	میرپی	زراندر
حیر	ادهر	سنارا	پرتزاد
ساج	جمین	ستمرد	شیرداد
فرا	کافک	نانده	آیکابر
صیت	صصلو	جهاعا	ستژوجع

جدول ۱- نمونه هایی از کلمات تولید شده به وسیله مدل پیشنهاد شده کلمات فارسی

به منظور بررسی کیفی کلمات تولید شده توسط مدل، تعدادی از کلماتی را که بدون استفاده از مدل و به شکل کاملا تصادفی تولید شده اند می آوریم: ضمچ، چضص، بططپ، ذضجژ، دلزنذ، اژخذ، رصۀمژن و ظمشخجض. ملاحظه میشود که این کلمات هیچ شباهتی به کلمات زبان فارسی ندارند و هیچ یک به سادگی قابل تلفظ نیستند. اما در مقابل، مدل پیشنهاد شده توانسته است کلمات خوش آهنگی را تولید کند و حتی در برخی موارد (سطر اول جدول) موفق به تولید کلماتی شده است که واقعا عضوی از مجموعه کلمات زبان فارسی میباشند اما در لغتنامه ۴۱۰۰۰ کلمه ای، که برای آموزش مدل نیز به کار رفت، موجود نمیباشند.

به عنوان کاربردی دیگر، با استفاده از مدل ارائه شده لگاریتم احتمال (Logprob) تعدادی کلمه (دنباله مشاهده) درست و نادرست محاسبه میشود:

Logprob(تعالی) = -12.9
Logprob(نغالی) = -16.9

Logprob(مهندسی) = -15.4
Logprob(موندشی) = -14.1
Logprob(مهنذسی) = -36.7

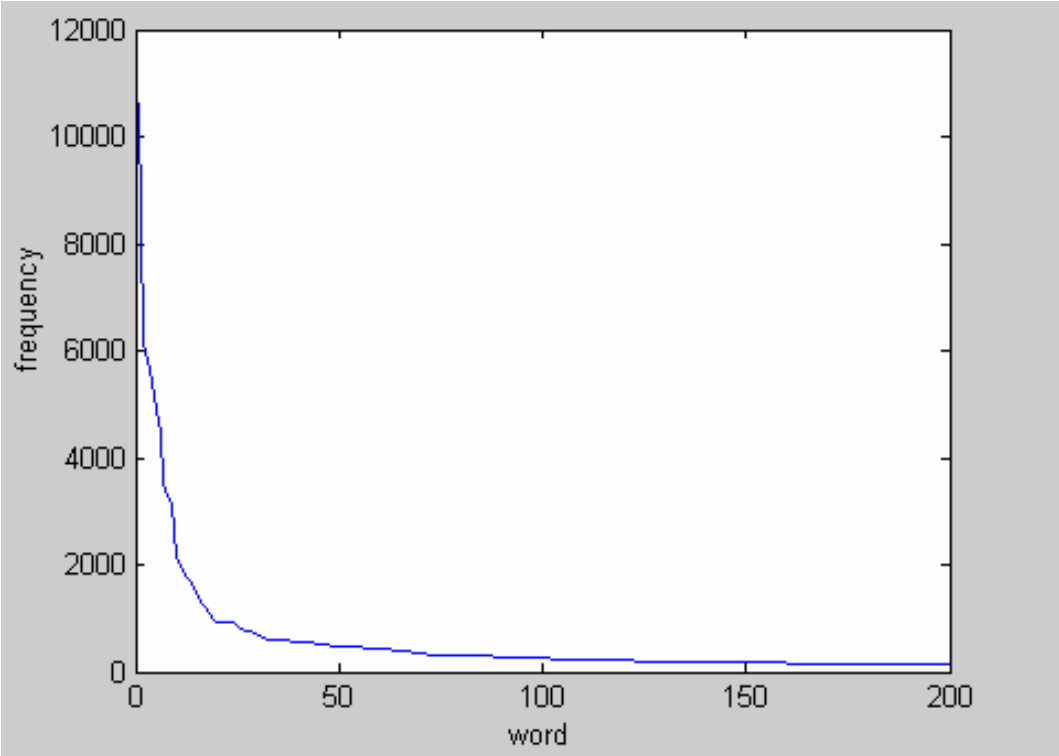
Logprob(ستژوجع) = -22.95
Logprob(ظمشخجض) = -44.7

واضح است که معیار Logprob برای یک کلمه هرچه نزدیک تر به صفر باشد، یعنی آن کلمه به مدل نزدیکتر است. همانطور که قبلا گفته شد، به دلیل ماهیت تغییر پذیر کلمات زبان، این امکان وجود ندارد که برای تشخیص کلمات نادرست (غیر متعلق به زبان) از یک لغتنامه ثابت استفاده کرد؛ یعنی تشخیص کلمات نادرست بر خلاف آنچه که ممکن است در ابتدا به نظر برسد مساله ساده ای نیست. اما با توجه به مثالهای بالا به نظر میرسد که با استفاده از مدل ارائه شده میتوان حتی بدون استفاده از لغتنامه، کلمات بسیار نادرست (بسیار نامحتمل) را تشخیص داد.

قسمت سوم

مدل آماری زبان

همانطور که بخشهای قبل ذکر شد، فرکانس یا احتمال وقوع کلمات یک زبان بسیار متفاوت است. این احتمالات برای زبانهای طبیعی از توزیعی به نام zipf تبعیت میکند. این توزیع برای ۲۰۰ کلمه اول (پر استفاده تر) زبان فارسی با استفاده از نوشته جات آموزشی ما در نمودار شکل ۳ نشان داده شده است.



شکل ۳- نمودار فراوانی ۲۰۰ کلمه اول زبان فارسی

با شمارش دفعات وقوع یک کلمه در یک مجموعه نوشته جات و تقسیم آن بر کل تعداد کلمات میتوان یک تخمین ML برای احتمال وقوع هر کلمه به دست آورد. این ایده را میتوان به زوج کلمات (و ترکیب های طولانی تر) تعمیم داد با این هدف که بعد از دیدن یک یا چند کلمه بتوان کلمه بعدی را حدس زد. پیش بینی یا حدس زدن کلمه بعدی یکی از اعمال ضروری در کاربردهای بازشناسی گفتار، بازشناسی متون (دستنویس یا تایپی) و تشخیص و تصحیح غلطهای املایی است که در آنها تشخیص کلمه کاری دشوار است زیرا ورودی مبهم و همراه با نویز زیاد است. و بنابراین در نظر گرفتن ورودی های قبلی میتواند اطلاعات مهمی درباره گزینه های ممکن برای ورودی جاری در اختیار بگذارد. در مدلهای آماری زبان هدف پیش بینی کلمه بعدی یا به بیان دیگر محاسبه احتمال دنباله کلمات (و جملات) است. روشهای محاسبه

(انتساب) احتمال به یک جمله میتوانند برای محاسبه احتمال کلمه بعدی در یک جمله ناقص (و بر عکس) به کار روند. دانستن احتمال یک جمله همچنین در کاربردهایی از جمله مشخص کردن نقش کلمات، رفع ابهام معنای کلمات (در ترجمه) و تجزیه احتمالی بسیار موثر است.
در مساله تصحیح غلطهای املایی ممکن است که یک یا چند اشتباه تایپی، کلمه مورد نظر را تبدیل به یک کلمه دیگر از زبان (یعنی یک کلمه درست و مجاز) کنند؛ واضح است که در این حالت استفاده از یک لغتنامه به تنهایی نمیتواند کمکی کند. اما با استفاده از اطلاعات کلمات مجاور و گرامرهای زبان در بیشتر موارد میتوان غلط را تشخیص داد و حتی تصحیح کرد. به عنوان مثال کلمه «دوستش» که در جمله «کتاب را به درستش داد» به اشتباه «درستش» تایپ شده است، ترکیب بسیار غیر محتمل «به درستش» را نتیجه داده است که با استفاده از یک گرامر آماری ساده نیز قابل تشخیص است. البته باید توجه کرد که بسیاری از کلمات و جملات کاملا درست و بامعنی زبان ممکن است احتمال وقوع پایینی داشته باشند، که این مساله تشخی خطا را دشوار میکند.
مدل زبان یا به بیان دیگر مدل پیش بینی کلمه ای که برای فارسی استفاده کرده ایم و در اینجا شرح میدهیم یک مدل آماری بسیار پر استفاده به نام N-gram است که در آن احتمال کلمه N اُم با استفاده از N-1 کلمه قبلی تخمین زده میشود. موفقیت این مدلها اولین بار در آزمایشگاه های بازشناسی گفتار IBM به اثبات رسید و پس از آن در بسیار از زمینه ها مورد توجه و استفاده محققین قرار گرفتند.

شمارش کلمات

برای تخمین احتمال نیاز به شمارش ترکیب ها داریم. همانگونه که قبلا ذکر شد، پردازش آماری زبان بر مبنای یک مجموعه بزرگ نوشته جات است. البته باید توجه کرد که در بعضی کاربردها - از جمله بررسی گرامر، تولید صدا یا شناسایی نویسنده - لازم است تا علائمی همچون '،' و '.' را نیز به عنوان کلمه در نظر گرفت و در شمارش کلمات و ترکیبات متوالی، آنها را نیز به حساب آورد. مساله دیگری که باید به آن توجه کرد اشکال مختلف کلمات است. در بیشتر سیستم های مبتنی بر N-gram اشکال مختلف یک کلمه به عنوان کلمات مجزا در نظر گرفته میشوند؛ یعنی هیچ تلاشی برای ریشه یابی انجام نمیشود که البته این ساده سازی در بعضی کاربردها مناسب نیست.

N-gram ساده

در این بخش مدل N-gram ساده یا هموار نشده (unsmoothed) را شرح میدهیم. یاد آوری میکنیم که مدلهایی که در اینجا برای دنباله کلمات در نظر میگیریم مدلهای احتمالاتی هستند یعنی روشهایی برای انتساب احتمال به رشته های کلمات که هم میتوانند برای

محاسبه احتمال یک دنباله کلی استفاده شوند و هم برای پیش بینی احتمالاتی کلمه بعـدی در یک دنباله.

در ساده ترین مدلِ ممکن برای دنباله کلمات، وقوع هر کلمه پس از هر کلمه دیگر مجاز اسـت. به بیان احتمالاتی، هر کلمه احتمال وقوع یکسانی پس از هر کلمه دلخواه دارد. به عنـوان مثـال بـا داشـتن لغتنامـه 41000 کلمـه ای، ایـن احتمـال برابـر اسـت بـا 41000 / 1 یـا تقریبـا 0.000024. در یک مدل نسبتا پیچیده تر فرض میکنیم باز هم هر کلمه میتوانـد پـس از هـر کلمه دلخواه قرار گیرد اما احتمال وقوع کلمه بعدی (کلمه دوم) برابـر اسـت بـا فراوانـی تکـرار نرمال (احتمال وقوع) آن. به عنوان مثال کلمه «این» در نوشته جات آموزشی ما فراوانـی تکـرار نرمال 0.0182 دارد، پس احتمال اینکه «این» بعد از هر کلمه بیاید 0.0182 اسـت. یعنـی در این مدل ساده که unigram نام دارد، کلمـات گذشـته در پـیش بینـی کلمـه بعـدی تـاثیری ندارند. در واقع با استفاده از فراوانی های نسبی میتوان یک توزیع احتمال روی کلمات بعدی بـه دست آورد. اما در نظر نگرفتن تاریخچه گذشته در بیشتر موارد نمیتواند پیش بینی های به درد خوری را نتیجه دهد. به عنوان مثال میدانیم که پس از عبارت «خیلی دیر» قرار گرفتن «شـده» بسیار معقول تر از «این» است، امـا مـدل سـاده unigram، «ایـن» را محتمـل تـر از «شـده» میداند، زیرا احتمال وقوعش به تنهایی بیشتر است. این مثال نشان میدهـد کـه احتمـال وقـوع یک کلمه را باید به شرط کلمات قبل از آن تخمین زد نه به طور مجزا. اگر در مثـال ذکـر شـده حتی تاریخچه ای به اندازه یک - یعنی یک کلمه گذشته - را در نظر بگیریم، مـدلی بـه دسـت می آید که ترکیب «خیلی دیر شده» را به «خیلی دیر این» ترجیح میدهد زیرا $P($ دیر $|$ شـده $)$ بیشتر از $P($ دیر $|$ این $)$ است.

با توجه به مطالب گفته شده، حالا روش محاسـبه احتمـال یـک رشـته کلمـات (کـه بـه شـکل $w_1 \ldots w_n$ یا w_1^n نمایش داده میشوند) را بررسی میکنیم. اگر فرض کنیم هر کلمه در موقعیـت درستش یک رویداد مستقل است، میتوانیم این احتمال را به شکل زیر نشان دهیم:

$$P(w_1, w_2, \ldots, w_{n-1}, w_n) \quad (۶۶)$$

با استفاده از رابطه زنجیری احتمال این رابطه را میتوان به شکل زیر تجزیه کرد:

$$P(w_1^n) = P(w_1)P(w_2|w_1)P(w_3|w_1^2)\ldots P(w_n|w_1^{n-1}) = \prod_{k=1}^{n} P(w_k|w_1^{k-1}) \quad (۶۷)$$

حالا سوال این است که چطور احتمالاتی همچون $P(w_n \mid w_1^n)$ محاسبه کنیم. به نظر میرسـد هیچ راه سر راستی برای محاسبه احتمال یک کلمه به شرط دنباله ای طولانی از کلمـات قبلـی وجود نداشته باشد، مگر اینکه یک مجموعه نوشته جات بسیار بسـیار بـزرگ در دسـترس باشـد که عملا این طور نیست. این مساله را با یک ساده سازی مفید حل میکنیم. یعنی احتمـال یـک کلمه را به شرط کلمات قبلی را تقریب میزنیم. در ساده ترین تقریب احتمـال یـک کلمـه را بـه

شرط تنها یک کلمه قبلی به دست می آوریم. این مدل ساده bigram نام دارد که در آن $P(w_n \mid w_1^n)$ با $P(w_n \mid w_{n-1})$ تقریب زده میشود.

این فرض که احتمال یک کلمه تنها به کلمه قبلی وابسته است، فرض مارکوف (مرتبه اول) است. همانطور که در مدل کلمات دیدیم، مدل های مارکوف دسته ای از مدلهای احتمالاتی هستند که در آنها فرض میکنیم که میتوانیم احتمال رویدادی در آینده را بدون در نظر گرفتن تاریخچه ای طولانی از رویدادهای گذشته پیش بینی کنیم. بنابراین مدل bigram پایه را میتوان به عنوان نوعی زنجیره مارکوف که یک حالت به ازای هر کلمه دارد در نظر گرفت.

بدیهی است که میتوانیم مدل bigram را، که تنها یک کلمه گذشته را در نظر میگیرد، به مدل trigram که دو کلمه گذشته را در نظر میگیرد و در حالت کلی به مدل N-gram که N-1 کلمه گذشته را در نظر میگیرد تعمیم دهیم. پس مدل bigram یک مدل مارکوف مرتبه اول، trigram مدل مارکوف مرتبه دوم و N-gram مدل مارکوف مرتبه N-1 اُم است. (در عمل مدلهای bigram یا حداکثر trigram استفاده میشوند).

معادله کلی برای تقریب N-gram - احتمال شرطی کلمه بعدی در یک دنباله - برابر است با:

$$P(w_n \mid w_1^{n-1}) \approx P(w_n \mid w_{n-N+1}^{n-1}) \quad (۶۸)$$

این رابطه نشان میدهد که احتمال کلمه w_n به شرط همه کلمات قبلی میتواند با احتمال این کلمه به شرط N کلمه قبلی تقریب زده شود.

بنابراین برای یک گرامر bigram میتوانیم احتمال یک رشته کامل را با جایگزین کردن رابطه ۶۸ در رابطه ۶۷ محاسبه کنیم:

$$P(w_1^n) \approx \prod_{k=1}^{n} P(w_k \mid w_{k-1}) \quad (۶۹)$$

البته باید توجه کرد که پیاده سازی مستقیم این روابط ممکن است منجر به خطای پاریز (underflow) شود زیرا هرکدام از احتمال ها معمولا عددی بسیار کوچکتر از ۱ است و حاصلضرب تعداد کمی از این اعداد هم از محدوده قابل نمایش در کامپیوتر تجاوز میکند. برای حل این مشکل، همانطور که قبلا نیز دیده اید، بجای ضرب احتمالات، لگاریتم آنها را با هم جمع میکنیم و در انتها معکوس لگاریتم نتیجه را حساب میکنیم. یعنی برای جلوگیری از خطای پاریز، بجای ذخیره و کار با احتمال، با لگاریتم آن کار میکنیم.

نکته دیگری که در اینجا لازم به توضیح است محاسبه احتمال یک کلمه در ابتدای جمله است. یعنی وقتی که کلمات گذشته ای وجود نداشته باشند. در این حالت میتوانیم به سادگی از مدل unigram استفاده کنیم اما برای به دست آوردن کارایی بهتر، فرض میکنیم شبه کلمه $<s>$ در ابتدای هر جمله (آموزشی / آزمایشی) وجود دارد کع برای محاسبه احتمال ها مانند کلمات عادی شمرده میشود. بنابراین احتمال کلمه w در ابتدای یک جمله با مدل bigram برابر

است با $P(w \mid <s>)$ و اگر از مدل trigram استفاده کنیم، احتمال کلمه اول جمله برابر است با $P(w \mid <s>, <s>)$ و به همین ترتیب.

آموزش مدلهای N-gram از طریق شمارش و نرمال کردن انجام میشود. در مدلهای احتمالاتی نرمال کردن به معنای تقسیم به یک مجموع است به طوریکه احتمالات نتیجه در محدوده ۰ و ۱ قرار گیرند. به عنوان مثال برای محاسبه (آموزش) یک مدل bigram ساده، با در دست داشتن یک مجموعه نوشته جات، تعداد بارهای تکرار هر bigram (زوج کلمه) را میشماریم و سپس آن را بر مجموع دفعات تکرار تمام bigram هایی که با کلمه اول شروع میشوند تقسیم میکنیم. یعنی:

$$P(w_n \mid w_{n-1}) = \frac{C(w_{n-1}w_n)}{\sum_w C(w_{n-1}w)} \quad (۷۰)$$

این رابطه را میتوان ساده کرد، زیرا مجموع همه bigram هایی که با کلمه w_{n-1} شروع میشوند برابر است با دفعات تکرار این کلمه (یعنی تعداد unigram های w_{n-1}). بنابراین:

$$P(w_n \mid w_{n-1}) = \frac{C(w_{n-1}w_n)}{C(w_{n-1})} \quad (۷۱)$$

و برای حالت کلی تخمین N-gram از رابطه زیر استفاده میشود:

$$P(w_n \mid w_{n-N+1}^{n-1}) = \frac{C(w_{n-N+1}^{n-1}w_n)}{C(w_{n-N+1}^{n-1})} \quad (۷۲)$$

این رابطه احتمال N-gram را با تقسیم کردن فراوانی مشاهده شده یک دنباله خاص بر فراوانی مشاهده شده پیش دنباله تخمین میزند. این نسبت فراوانی نسبی نامیده میشود. استفاده از فراوانی نسبی به عنوان روشی برای تخمین احتمال مثالی از کاربرد تکنیک MLE است؛ زیرا مجموعه پارامترهای نتیجه شده، احتمال مجموعه آموزشی T به شرط مدل M (یعنی $P(T \mid M)$) را ماکزیمم میکند.

البته به جای استفاده از فراوانی های نسبی روشهای بهتری برای تخمین احتمالات N-gram وجود دارد که در بخش بعد آنها را بررسی میکنیم. اما آن روشهای پیشرفته نیز به نوعی از ایده فراوانی نسبی استفاده میکنند.

مثال

جدول ۲ فراوانی تکرار bigram های زیر مجموعه ای از نوشته جات آموزشی ما را نشان میدهد که شامل ۱۶۱۶ نوع کلمه و تقریبا ۱۰۰۰۰ جمله است. همانطور که ملاحظه میکنید بیشتر مقادیر صفر هستند. در واقع چون تنها هفت کلمه را انتخاب کرده ایم، این ماتریس

خلوت تر نیز شده است. فراوانی تکرار این هفت کلمه به این شـرح اسـت: او: ۳۴۳۷، میخواهـد: ۱۲۱۵، در: ۳۲۵۶، مسابقات: ۹۳۸، کشتی: ۲۱۳، شرکت: ۱۵۰۶، کند: ۴۵۹.

	او	میخواهد	در	مسابقات	کشتی	شرکت	کند
او	8	1087	0	13	0	0	0
میخواهد	3	0	786	0	6	8	6
در	3	0	10	860	3	0	12
مسابقات	0	0	2	0	19	2	52
کشتی	2	0	0	0	0	120	1
شرکت	19	0	17	0	0	0	0
کند	4	0	0	0	0	1	0

جدول ۲- فراوانی تکرار bigram های زیر مجموعه ای از نوشته جات آموزشی

جدول ۳ احتمالات bigram ها را پس از نرمال سازی - یعنی تقسیم هر سطر بر فراوانی تکـرار unigram مناسب - نشان میدهد.

	او	میخواهد	در	مسابقات	کشتی	شرکت	کند
او	.0023	.32	0	.0038	0	0	0
میخواهد	.0025	0	.65	0	.0049	.0066	.0049
در	.00092	0	.0031	.26	.00092	0	.0037
مسابقات	0	0	.0021	0	.020	.0021	.055
کشتی	.0094	0	0	0	0	.56	.0047
شرکت	.013	0	.011	0	0	0	0
کند	.0087	0	0	0	0	.0022	0

جدول ۲- فراوانی تکرار bigram های زیر مجموعه ای از نوشته جات آموزشی

چند ویژگی مدلهای N-gram

از ویژگی های قابل انتظار مدلهای N-gram این است که دقت (کارایی) مدل با افزایش مقـدار N افزایش می یابد. علیرغم ایـن واقعیـت، عمـلا در بیشـتر کاربردهـا از مـدلهای bigram یـا حداکثر trigram استفاده میشود؛ زیرا مدلهای مرتبه بالاتر از ۳ برای آموزشِ مناسـب، احتیـاج به مجموعه نوشته جات بسیار بزرگی دارند و در غیر این صورت نمیتوان تخمین هـای مناسـبی برای احتمالات به دست آورد. علاوه بر این، مدلهای مرتبـه بـالاتر از ۳ بسـیار بـزرگ هسـتند و ذخیره و استفاده از آنها احتیاج به حافظه زیادی دارد. مرتبه حافظـــــه مـورد نیـاز یـک مـدل N-gram برابر است با تعداد ترکیبهای مختلف N کلمه ای که دارای حد بالای V^N اسـت (V اندازه لغتنامه مورد استفاده است).

ویژگی دیگر مدلهای N-gram وابستگی زیاد آنها به نوشته جات (به خصوص نوع و اندازه) آموزشی است. یکی از روشهای معمول برای مشاهده عملکرد کیفی یک مدل N-gram، تولید رشته های تصادفی کلمات با استفاده از مدل است. به این ترتیب که اولین کلمه بر اساس احتمال unigram انتخاب میشود و سپس دومین کلمه با مدل bigram و پس از آن کلمات بعدی میتوانند با مدل bigram یا trigram تولید شوند.

نظر به اینکه احتمالات در یک مدل آماری همچون N-gram از یک مجموعه آموزشی استخراج میشوند، این مجموعه را باید با دقت انتخاب کرد. اگر مجموعه آموزشی تنها مربوط به یک زمینه (موضوع) خاص باشد، احتمالات نتیجه شده نمی توانند برای جملات جدید به شکلی مناسب تعمیم یابند. از طرفی اگر مجموعه نوشته جات آموزشی بسیار عمومی و کلی باشد، ممکن است احتمالات نتیجه شده برای آن کاربرد خاص مناسب نباشند.

برای آموزش و سپس محاسبه کارایی یک مدل، همانند سایر مسائل یادگیری محاسباتی، مجموعه نوشته جات موجود را به دو مجموعه مجزای آموزشی و آزمایشی تقسیم میکنیم؛ مدل را بر اساس مجموعه آموزشی میسازیم و سپس کارایی آن را با استفاده از معیاری به نام سرگشتگی (Perplexity) - که معیاری مشابه با آنتروپی است - روی مجموعه آزمایشی محاسبه میکنیم. البته در برخی موارد به بیش از یک مجموعه آزمایشی احتیاج داریم. مثلا فرض کنید که چند مدل مختلف را در اختیار داریم و هدف این است که بهترین را انتخاب و سپس کارایی آن را محاسبه کنیم. البته باید توجه کرد که برای مقایسه کارایی این مدلها لازم است از تست های آماری استفاده شود تا معلوم شود تفاوت بین دو مدل تا چه حد قابل توجه است.

هموارسازی

مساله اصلی مدلهای استاندارد N-gram که تا حالا بررسی کردیم، این است که تخمین های احتمال آنها با استفاده از یک مجموعه نوشته جات آموزشی "محدود" به دست می آیند که به هر حال نمیتواند شامل تمام ترکیبهای (N-gram های) مجاز و با معنی زبان باشد. یعنی مثلا ماتریس bigram با هر مجموعه آموزشی که محاسبه شود، ماتریسی خلوت است. یعنی شامل تعداد بسیار زیادی bigram با احتمال صفر است که بخشی از این bigram واقعا مجاز و بامعنی هستند و لذا باید احتمال غیر صفری داشته باشند. این مشکل را در مثال جدول ۳ ملاحظه میکنید، به عنوان نمونه در سطر اول، ترکیب «او شرکت» علیرغم معقول بودن، احتمالی صفر دارد (یعنی از دید مدل غیر ممکن تلقی میشود).

حتی اگر مشکل فراونی های صفر وجود نداشته باشد، روش MLE وقتی مقدار فراوانی ها کوچک باشد، باز هم تخمین های ضعیفی را نتیجه میدهد. برای حل این مشکلات از تکنیک های هموارسازی (smoothing) استفاده میشود که در آنها به احتمالات صفر (و پایین)

مقادیر غیر صفری (و بیشتری) منسوب میشود و بـه همـین نسـبت از احتمـالات زیـاد کاسـته میشود (تا جمع احتمالات ۱ باقی بمانـد). در دو بخـش زیـر، دو الگـوریتم (سـاده و پیشـرفته) هموارسازی را به همراه مثال شرح میدهیم.

هموارسازیِ جمع با یک

اولین و ساده ترین روشی که برای هموارسازی - یا اجتنـاب از فراوانـی هـای صـفر - بـه ذهـن میرسد، روش «جمع با یک» است و اگرچه که کـارایی خـوبی نـدارد و در عمـل چنـدان مـورد استفاده قرار نمیگیرد اما مفاهیم کلی و اساسی هموارسازی را در بر دارد و بررسی آن به درک و توصیف الگوریتم های پیچیده تر کمک میکند.

در ابتدا به منظور سادگی، کاربرد روش جمع با یک را برای هموارسـازی احتمـالات unigram شرح میدهیم. تخمین ML معمولی ناهموار، با تقسیم کردن دفعات تکرار یک کلمـه بـر تعـداد کل کلـمات موجود در مجموعه نوشته جات آموزشی (که با N نمایش داده میشـود) بـه دسـت می آید:

$$P(w_x) = \frac{c(w_x)}{\sum_i c(w_i)} = \frac{c(w_x)}{N}$$

روشهای مختلف هموارسازی بر مبنای یک فراوانی تصحیح شـده c^* هسـتند. تصـحیح فراوانـی برای روش جمع با یک اینطور تعریف میشود: مقدار فراوانـی پـس از جمـع بـا یـک، در ضـریب نرمال سازی $\frac{N}{N+V}$ ضرب میشود که V اندازه لغتنامه (یا تعداد نـوع کلمـات، یـا بـه عبـارتی دیگر تعداد unigram های منحصر به فرد) است. چون به هر کلمـه (نـوع کلمـه) یکـی اضـافه میکنیم، به اندازه کل کلمات موجود به اندازه V اضافه میشود و بنابراین فراوانـی تصـحیح شـده برای روش هموارسازی جمع با یک اینطور تعریف میشود:

$$c_i^* = (c_i + 1)\frac{N}{N+V} \qquad (۷۳)$$

و این فراوانی ها با تقسیم (نرمال) کردن بر N تبدیل به احتمال میشوند.

در واقع ایده اصلی همه الگوریتم های هموارسازی کاستن از فراوانـی هـای غیـر صـفر (و دارای مقدار زیاد) با هدف انتساب فراوانی (و در نتیجه احتمال) غیر صفر به فراوانی هـای صـفر اسـت. بنابراین، بجای استفاده از فراوانی های کاهش یافته c^*، برخـی محققـین تـرجیح میدهنـد کـه الگوریتم های همواره سازی را بر حسب یک ضریب کاهش d_c تعریف کنند کـه برابـر اسـت بـا نسبت فراوانی کاهش یافته به فراوانی اصلی: $d_c = \frac{c^*}{c}$.

و به عنوان روشی دیگر، میتوانیم احتمالات p^* را مستقیما از فراوانی های اصلی حساب کنیم:

$$p_i^* = \frac{c_i + 1}{N + V}$$

حال که روش جمع با یک را برای unigram توضیح دادیم، آنرا به bigram مثال قبل (جدول ۲) اعمال میکنیم. جدول ۴ فراوانی های با یک جمع شده (هموار شده) و جدول ۵ احتمالات مربوطه را نشان میدهد.

	او	میخواهد	در	مسابقات	کشتی	شرکت	کند
او	9	1088	1	14	1	1	1
میخواهد	4	1	787	1	7	9	7
در	4	1	11	861	4	1	13
مسابقات	1	1	3	1	20	3	53
کشتی	3	1	1	1	1	121	2
شرکت	20	1	18	1	1	1	1
کند	5	1	1	1	1	2	1

جدول ۴- فراوانی های تکرار همواره شده مثال جدول ۲ با روش جمع با یک

	او	میخواهد	در	مسابقات	کشتی	شرکت	کند
او	.0018	.22	.00020	.0028	.00020	.00020	.00020
میخواهد	.0014	.00035	.28	.00035	.0025	.0032	.0025
در	.00082	.00021	.0023	.18	.00082	.00021	.0027
مسابقات	.00039	.00039	.0012	.00039	.0078	.0012	.021
کشتی	.0016	.00055	.00055	.00055	.00055	.066	.0011
شرکت	.0064	.00032	.0058	.00032	.00032	.00032	.00032
کند	.0024	.00048	.00048	.00048	.00048	.00096	.00048

جدول ۵- احتمالات هموار شده مثال جدول ۲ با روش جمع با یک

یادآوری میکنیم که احتمالات bigram با نرمال سازی هر سطر با فراوانی unigram محاسبه میشود (رابطه ۷۱). برای فراوانی های هموار شده (با روش جمع با یک) لازم است فراوانی هر unigram را به اندازه تعداد کلمات منحصر به فرد (اندازه لغتنامه که با V نشان داده میشود) افزایش دهیم:

$$P(w_n \mid w_{n-1}) = \frac{C(w_{n-1}w_n) + 1}{C(w_{n-1}) + V} \quad (۷۴)$$

چون در این مثال $V = 1616$ است، فراوانی های unigram (تک کلمه ها) به این شکل افزایش می یابند:

او: ۳۴۳۷ + ۱۶۱۶ = ۵۰۵۳ میخواهد: ۱۲۱۵ + ۱۶۱۶ = ۲۸۳۱

در: ۴۸۷۲ = ۱۶۱۶ + ۳۲۵۶ مسابقات: ۲۵۵۴ = ۱۶۱۶ + ۹۳۸

کشتی: ۱۸۲۹ = ۱۶۱۶ + ۲۱۳ شرکت: ۳۱۲۲ = ۱۶۱۶ + ۱۵۰۶

کند: ۲۰۷۵ = ۱۶۱۶ + ۴۵۹

و نتیجه که احتمالات هموار شده است در جدول ۵ نشان داده شد.

برای اینکه ببینیم یک الگوریتم هموار سازی چگونه فراوانی های اصلی را تغییر داده است، معمولا ماتریس فراوانی را بازسازی میکنیم. فراوانی های تصحیح شده که با رابطه ۷۳ محاسبه میشوند برای این مثال در جدول ۶ نشان داده شده اند. توجه کنید که این روش (جمع با یک) فراوانی ها را به شدت تغییر داده است. مثلا (میخواهد در)C از ۷۸۶ به ۳۳۳ کاهش یافته است و در فضای احتمال نیز (میخواهد | در)P از 65. (در حالت ناهموار) به 28. (در حالت هموار) کم شده است. ضریب کاهش d_c نیز نشان میدهد که فراوانی برای هر کلمه اول چند برابر شده است. مثلا در اینجا bigram هایی که با «کشتی» شروع شده اند، تقریبا ۸ برابر کم شده اند.

	او	میخواهد	در	مسابقات	کشتی	شرکت	کند
او	6	740	.68	10	.68	.68	.68
میخواهد	2	.42	331	.42	3	4	3
در	3	.69	8	594	3	.69	9
مسابقات	.37	.37	1	.37	7.4	1	20
کشتی	.36	.12	.12	.12	.12	15	.24
شرکت	10	.48	9	.48	.48	.48	.48
کند	1.1	.22	.22	.22	.22	.44	.22

جدول ۶- فراوانی های بازسازی شده مثال جدول ۲ با روش جمع با یک

تغییرات شدید مقادیر فراوانی و احتمال در اثر جابجایی زیاد توزیع احتمال به طرف احتمالات صفر (و نزدیک به صفر) میباشد. مشکل از اینجا ناشی میشود که مقدار ۱ برای جمع با هر فراوانی به طور اختیاری انتخاب شده است. البته میتوان با جمع کردن مقادیر کوچکتر (مثلا نیم یا یک هزارم) تا حدودی از این مشکل اجتناب کرد؛ اما برای انتخاب این عدد هم باید معیاری معقول وجود داشته باشد. به طور کلی هموارسازیِ جمع با یک روشی ضعیف است و در مقایسه با روشهای پیشرفته تر عملکرد بسیار بدتری دارد. همچنین تحقیقات نشان داده اند که واریانس های فراوانی های تولید شده با روشِ جمع با یک واقعا بدتر از چیزی است که با روش MLE معمولی (ناهموار) به دست آید.

هموارسازی Witten-Bell

روش هموار سازی یا کاهش Witten-Bell تنها کمی پیچیده تر از روش جمع با یک است، اما کارایی بسیار بهتری دارد و یکی از روشهای پراستفاده در سیستم های بازشناسی گفتار

است. از روشهای پراستفاده دیگر میتوان از Good-Turing و Katz نام برد. روش Witten-Bell بر پایه یک ایده ساده اما زیرکانه درباره رویدادهای با فراوانی صفر (مشاهده نشده) است: یک رویداد یا N-gram با فراوانی صفر وقتی اتفاق می افتد اولین باری است که آن را مشاهده میکنیم؛ پس احتمال دیدن یک N-gram با فراوانی صفر میتواند با احتمال دیدن آن برای اولین بار مدل شود؛ که این یک مفهوم تکرار شونده در پردازش آماری زبان است. ایده اصلی این است که از فراوانی چیزهایی که تنها یک بار دیده شده اند (رخ داده اند) برای تخمین فراوانی چیزهایی که تا حالا دیده نشده اند (رخ نداده اند) استفاده کنیم. این ایده در برخی الگوریتمهای دیگر هموارسازی و روشهای تشخیص نقش کلمه نیز استفاده شده است.
برای محاسبه احتمال دیدن یک N-gram برای اولین بار باید دفعاتی را که برای اولین بار در مجموعه نوشته جات مشاهده میشود حساب کرد؛ و این به سادگی قابل محاسبه است زیرا تعداد N-gram های اولین بار (جدید) یعنی تعداد N-gram های منحصر به فرد در مجموعه داده ها. بنابراین مجموع احتمال همه N-gram های دارای فراوانی صفر را با تعداد انواع N-gram ها تقسیم بر مجموع تعداد کل و تعداد انواع N-gram ها تخمین میزنیم:

$$\sum_{i:c_i=0} p_i^* = \frac{T}{N+T} \quad (۷۵)$$

دلیل اینکه با مجموع تعداد کل و تعداد انواع N-gram ها نرمال میکنیم این است که مجموعه نوشته جات را میتوانیم به عنوان یک سری از رویدادها در نظر بگیریم: یک رویداد برای هر کلمه و یک رویداد برای هر نوع جدید. بنابراین رابطه ۷۵ در واقع یک تخمین ML برای رخ دادن یک رویداد جدید است. باید توجه کرد که تعداد انواع مشاهده شده T با تعداد کل (یا اندازه لغتنامه) V که قبلا در روش هموارسازی جمع با یک استفاده شد تفاوت دارد. T تعداد انواعی است که تا کنون دیده ایم، در حالیکه V تعداد "همه انواع ممکن" است.
رابطه ۷۵ مجموع احتمال N-gram های دیده نشده است که باید بین همه N-gram های دارای فراوانی صفر تقسیم شود. میتوانیم این تقسیم را به طور مساوی انجام دهیم. فرض کنید Z تعداد N-gram های منحصر به فرد با فراوانی صفر باشد، حالا هر unigram صفر سهم مساوی از توزیع احتمال میگیرد:

$$Z = \sum_{i:c_i=0} 1 \quad (۷۶)$$

$$p_i^* = \frac{T}{Z(N+T)} \quad (۷۷)$$

احتمال مجموع N-gram های صفر، که از رابطه ۷۵ حساب میشود، با کاهش دادن احتمال N-gram های دیده شده تامین میشود:

(۷۸)
$$p_i^* = \frac{c_i}{N+T} \quad \text{if } c_i > 0$$

به بیان دیگر، فراوانی های هموار شده را میتوانیم مستقیما از رابطه زیر محاسبه کنیم:

(۷۹)
$$c_i^* = \begin{cases} \frac{T}{Z}\frac{N}{N+T}, & \text{if } c_i = 0 \\ c_i \frac{N}{N+T}, & \text{if } c_i > 0 \end{cases}$$

همانطور که ملاحظه میکنید، این روش هموارسازی برای unigram ها بسیار شبیه به روش جمع با یک است. اما وقتی معادلات به bigram تعمیم داده میشوند، تفاوت بزرگی می بینیم که به این دلیل است که در اینجا فراوانی انواع به تاریخچه ای وابسته است. برای محاسبه احتمال یک bigram تا حالا مشاهده نشده $w_{n-1}w_{n-2}$ از احتمال مشاهده bigram جدیدی که با w_{n-1} شروع میشود استفاده میکنیم. این یعنی تخمینی که برای bigram های جدید میزنیم به یک تاریخچه کلمه وابسته میشود. کلماتی که در تعداد کمتری از bigram ها اتفاق می افتند، تخمین bigram مشاهده نشده کمتری نسبت بر کلمات فراوان تر ارائه میکنند. این واقعیت را با وابسته کردن T (تعداد انواع bigram ها) و N (تعداد کل bigram ها) به w_x (کلمه قبلی) نشان میدهیم:

(۸۰)
$$\sum_{i:c(w_x w_i)=0} p^*(w_i \mid w_x) = \frac{T(w_x)}{N(w_x)+T(w_x)}$$

دوباره باید این جمع احتمال را بین همه bigram های دیده نشده توزیع کنیم. فرض کنید Z تعداد bigram های منحصر به فرد با فراوانی صفر باشد، حالا هر bigram صفر سهم مساوی از توزیع احتمال میگیرد:

(۸۱)
$$Z(w_x) = \sum_{i:c(w_x w_i)=0} 1$$

(۸۲)
$$p^*(w_i \mid w_{i-1}) = \frac{T(w_{i-1})}{Z(w_{i-1})(N+T(w_{i-1}))} \quad \text{if } c_{w_{i-1}w_i} = 0$$

مانند bigram های غیر صفر، با وابسته کردن T به تاریخچه، آنها را به همان ترتیب کاهش میدهیم:

(۸۳)
$$\sum_{i:c(w_x w_i)>0} p^*(w_i \mid w_x) = \frac{c(w_x w_i)}{c(w_x)+T(w_x)}$$

از روش کاهش Witten-Bell میتوان به شکل دیگری نیز استفاده کرد. در رابطه ۸۰ احتمالات bigram هموار شده را به کلمه قبلی وابسته کردیم. یعنی $T(w_x)$ (تعداد انواع

bigram ها) و $N(w_x)$ (تعداد کل bigram ها) را بـه w_x (کلمـه قبلـی) وابسـته کـردیم. امـا بجای این کار، میتوان هر bigram را به عنوان یک رویداد تکی در نظر گرفت بـدون توجـه بـه اینکه از دو کلمه تشکیل شده است. آنگـاه T تعـداد انـواع همـه bigram هـا و N تعـداد کـل bigram های اتفاق افتاده است. با این کـار در واقـع بجـای کـاهش دادن بـا احتمـال شـرطی $P(w_i|w_x)$ داریم از احتمال مشترک $P(w_iw_x)$ استفاده میکنیم. در اینجـا بـا $P(w_iw_x)$ دقیقـا همانند یک احتمال unigram برخورد میکنیم. البته این روش کاهش کمتر از روش اسـتاندارد (رابطه ۸۰) به کار رفته است.

منابع

Corazza, A., De Mori, R., Gretter, R. and Satta, G. (October 1993). "Language modeling using stochastic context-free grammars", *Speech Communication*, vol. 13(1-2), pp. 163-170.

Dugad, R. and Desai, U. B. (May 1996). "A Tutorial on Hidden Markov Models", *Technical Report No. SPANN-96.1*, Indian Institute of Technology, Bombay, India.

Juang, B. H. and Rabiner, L. R. (Sep. 1990). "The Segmental K-Means Algorithm for Estimating the Parameters of Hidden Markov Models", *IEEE Trans. on Acoustics, Speech and Signal Processing*, vol. 38(9), pp. 1639-1641.

Katz, S. M. (March 1987). "Estimation of Probabilities from Sparse Data for the Language Model Component of a Speech Recognizer", *IEEE Trans. on ASSP*, vol. 35(3), pp. 400-401.

Klatt, D. H. (1987). "Review of text-to-speech conversion for English", *Journal of the Acoustical Society of America*, vol. 82(3), pp. 737-793.

Lu, Z., Bazzi, I., Kornai, A., Makhoul, J., Natarajan, P. and Schwartz, R. (1999). "A Robust Language-Independent OCR System", *Proceedings of 27th AIPR Workshop: Advances in Computer-Assisted Recognition, SPIE Proceedings*.

Manning, C. D. and Schütze, H. (1999). *Foundations of Statistical Natural Language Processing*, The MIT Press.

Mitchell, Tom M. (1997). Machine Learning, McGraw-Hill.

Rabiner, L. R. (1989). "A Tutorial on Hidden Markov Models and Selected Applications in Speech Recognition", *IEEE Proceedings*, vol. 77(2), pp. 257-286.

Rosenfeld, R. (2000). "Two decades of statistical language modeling: Where do we go from here?", *Proceedings of the IEEE*, vol. 88, pp. 1270-1278.

Rosenfeld, R., Chen, S. F. and Zhu, X. (January 2001). "Whole-sentence exponential language models: a vehicle for linguistic-statistical integration", *Computer Speech & Language*, vol. 15(1), pp. 55-73.

Vinciarelli, A., Bengio, S. and Bunke, H. (June 2004). "Offline Recognition of Unconstrained Handwritten Texts Using HMMs and Statistical Language Models", *IEEE Trans. on PAMI*, vol. 26(6), pp. 709-720.

Viterbi, A. J. (1967). "Error Bounds for Convolutional Codes and an Asymptotically Optimum Decoding Algorithm", *IEEE Trans. on Information Theory*, *IT*-13(2), pp. 260-269.

Werner, S., Eichner, M., Wolff, M. and Hoffmann, R. (July 2004). "Toward Spontaneous Speech Synthesis - Utilizing Language Model Information in TTS", *IEEE Trans. on Speech and Audio Processing*, vol. 12(4), pp. 436-445.