
بسم الله الرحمن الرحیم

وزارت علوم، تحقیقات و فناوری

پژوهشگاه علوم و فناوری اطلاعات ایران (ایرانداک)

طرح پژوهشی

ارائه یک ماژول نرم‌افزاری در راستای ایجاد شبکه استنادی پارساهای موجود در پایگاه اطلاعات علمی ایران (گنج)

مجری

نصراله پاک‌نیت

همکاران

جلال‌الدین نصیری

مهرماه ۱۳۹۹

۳۳/۶۷۷۸
۱۳۹۹/۰۸/۰۳
ندارد

بنام خدا

ایرانداک در نیم‌قرنِ دوم کارِ خود همچنان جوان و کوشا، همراه هر پژوهش، پژوهشگر، و میاست‌گذار
۱۳۶۷ – ۱۳۹۹

گواهی

انجام طرح پژوهشی:

- **عنوان:** ارائه یک ماژول نرم‌افزاری در راستای ایجاد شبکه استنادی پارساهای موجود در پایگاه اطلاعات علمی ایران (گنج)
- **مجری:** دکتر نصراله پاک‌نیت
- **همکار:** دکتر جلال‌الدین نصیری
- **شمارهٔ قرارداد:** ۳۳/۴۷۷۲
- **تاریخ قرارداد:** ۱۳۹۸/۵/۶
- **تاریخ شروع:** ۱۳۹۸/۵/۶
- **تاریخ پایان:** ۱۳۹۹/۷/۲۸
- **ناظران:** دکتر آزاده محبی (ناظر محتوا)، مهندس مجتبی زالی (ناظر فنی)

در پژوهشگاه علوم و فناوری اطلاعات ایران گواهی می‌شود. گزارش پایانی این طرح، بر پایهٔ داوری در نشست ۳۸۹ (تاریخ ۱۳۹۹/۷/۳۰) شورای پژوهشی پژوهشگاه به تصویب رسیده است. لازم می‌دانم مراتب قدردانی خود را به مجری، همکار، و ناظران این طرح تقدیم دارم.

با آرزوی بهروزی
پرویز شهریاری
معاون پژوهش و آموزش

ارائه یک ماژول نرم‌افزاری در راستای ایجاد شبکه استنادی پارساهای موجود در پایگاه اطلاعات علمی ایران (گنج)

مدیر طرح پژوهشی: نصراله پاک‌نیت، پژوهشگاه علوم و فناوری اطلاعات ایران.

نشانی: تهران، خیابان انقلاب، چهارراه فلسطین، شماره ۱۰۹۰، طبقه چهارم، اتاق ۴۰۳

صندوق پستی: ۱۳۱۵۷۷۳۳۱۴ تلفن: ۶۶۴۹۴۹۸۰-۰۲۱(داخلی ۴۲۲)

وبگاه: http://irandoc.ac.ir/u/n-pakniat

رایانامه: pakniat@irandoc.ac.ir

همکاران: جلال‌الدین نصیری

چکیده

ارجاعات، که در مدارک علمی به وسیله رشته‌های مرجع در انتهای مدارک آورده می‌شوند، را می‌توان به عنوان ابزاری برای یافتن اطلاعات مورد علاقه به‌کار برد. همچنین می‌توان از آن‌ها به عنوان معیاری برای بررسی تأثیر و اهمیت یک مدرک علمی و در سطحی بالاتر، به عنوان معیاری برای بررسی صلاحیت پژوهشگران برای ارتقا، اعطای پژوهانه و پروژه‌های تحقیقاتی استفاده کرد. استفاده از ارجاعات با اهداف ذکر شده نیازمند ساخت شبکه تحلیل استنادی است تا روابط بین موجودیت‌هایی مانند اسناد، پژوهشگران، نشریات، دانشگاه‌ها و پژوهشگاه‌ها، انتشارات و ... مشخص گردد. ساخت شبکه تحلیل استنادی بر روی مجموعه‌ای از اسناد علمی به صورت سنتی (دستی) فرایندی بسیار زمان‌بر بوده و برای مجموعه‌های بزرگ از مدارک ممکن نیست. در نتیجه، به روش‌هایی نیازمندیم که بتوانند با ورودی مجموعه‌ای از مدارک علمی، به طور خودکار شبکه تحلیل استنادی متناظر با آن را ایجاد کنند.

پایگاه اطلاعات علمی ایران (گنج) از مهم‌ترین منابع علمی زبان فارسی بوده که توسط پژوهشگاه علوم و فناوری اطلاعات ایران (ایرانداک) توسعه یافته و دربرگیرنده اغلب پارساهای خاتمه‌یافته پژوهشگران ایرانی است. با توجه به اهمیت این پایگاه اطلاعاتی و عدم وجود روش‌هایی برای ساخت خودکار شبکه‌های تحلیل استنادی در زبان فارسی، در این پژوهش به مساله ساخت خودکار شبکه‌های تحلیل استنادی در پارساهای فارسی‌زبان می‌پردازیم. در این راستا، در ابتدا مساله ساخت خودکار شبکه تحلیل استنادی را به زیرمساله‌هایی چون استخراج مراجع از تمام‌متن، تعیین نوع مراجع، تجزیه مراجع و بررسی تطابق مراجع و مولفه‌های آن‌ها تقسیم نموده و در ادامه، با در نظر گرفتن این مسائل به عنوان مسائلی از جنس دسته‌بندی، از روش‌های یادگیری ماشینی و الگوریتم‌های تطبیق رشته تقریبی برای حل آن‌ها استفاده می‌کنیم. لازم به ذکر است که با توجه به فراوانی ارجاعات به منابع انگلیسی زبان در مدارک علمی فارسی، راه‌کارهای ارائه شده علاوه بر مراجع فارسی‌زبان، مراجع انگلیسی‌زبان را نیز در نظر می‌گیرند. روش یادگیری ماشینی استفاده شده در این پژوهش، روش بردار ماشین پشتیبان بوده که از نوع با نظارت است. با توجه به نیاز این روش‌ها به مجموعه داده‌هایی برچسب خورده و عدم وجود چنین داده‌هایی برای زبان فارسی، در این پژوهش چندین مجموعه داده برای مسائل موردنظر ایجاد می‌شود.

پس از ارائه راه‌کار برای زیر مساله‌های موردنظر، آن‌ها را با یکدیگر ترکیب کرده و تحت یک ماژول نرم‌افزاری ارائه می‌کنیم. ماژول ارائه شده، با دریافت تمام‌متن و فراداده‌های متناظر مجموعه‌ای از پارساها، شبکه تحلیل استنادی مربوطه را ایجاد کرده و در قالب Json

خروجی می‌دهد. نتایج ارزیابی راه‌کار نهایی ایجاد شده نشان‌دهنده مقادیر ۱ به عنوان پارامتر دقت، ۰,۷۸ به عنوان پارامتر فراخوانی و ۰,۸۸ به عنوان پارامتر F1 در بازیابی و تشخیص تطابق رشته‌های مرجع است.

کلیدواژه‌ها: شبکه تحلیل استنادی، نمایه‌سازی استنادی، تجزیه رشته‌های مرجع، تطبیق رشته‌های مرجع، دسته‌بندی، یادگیری ماشینی، ماشین بردار پشتیبان، فاصله لون‌اشتاین.

فهرست مطالب

عنوان **شماره صفحه**

فهرست جدول‌ها

عنوان **شماره صفحه**

فهرست شکل‌ها

عنوان **شماره صفحه**

قدردانی

اکنون‌که به یاری خدا طرح " ارائه یک ماژول نرم‌افزاری در راستای ایجاد شبکه استنادی پارساهای موجود در پایگاه اطلاعات علمی ایران (گنج)" به پایان رسیده است، بر خود لازم می‌دانم:

از ناظر گرامی طرح، سرکار خانم دکتر محبی، که بیش از ناظر و به مانند یک راهنما در انجام این پژوهش از مشاوره‌های ایشان بهره برده‌ام،

از جناب آقای مهندس زالی، که زحمت نظارت فنی این طرح را بر عهده داشته‌اند،

از سرکار خانم فاطمه جعفری که زحمت تبدیل برنامه نوشته شده در این پژوهش به زبان درخواستی پژوهشگاه را بر عهده داشته‌اند،

از همکاران پژوهشکده علوم اطلاعات به‌ویژه سرکار خانم دکتر پورنقی و جناب آقای دکتر سمائی، روسای گرامی فعلی و سابق پژوهشکده علوم اطلاعات، که همواره از حمایت این بزرگواران بهره برده‌ام، و

از جناب آقای دکتر علیدوستی، ریاست محترم پژوهشگاه، به پاس حمایت همیشگی از پژوهشگران به‌ویژه اینجانب

کمال تشکر را داشته باشم.

فصل اول:

مقدمه و اهداف

۱-۱- مقدمه

ارجاعات، که در مدارک علمی به وسیله رشته‌های مرجع (معمولا) در انتهای مدارک آورده می‌شوند، نقش مهمی در کتابخانه‌های دیجیتالی انتشار علمی مانند CiteSeer، arXiv e-Print، DBLP، Google Scholar و Scopus ایفا می‌کنند. علاوه بر استفاده از ارجاعات به عنوان ابزاری برای یافتن اطلاعات مورد علاقه، از این داده‌ها به عنوان معیاری برای بررسی تأثیر و اهمیت یک مقاله مشخص نیز استفاده می‌شود. در سطحی بالاتر، از میزان ارجاعات انجام شده به تحقیقات پژوهشگران به عنوان معیاری برای بررسی صلاحیت آن‌ها برای ارتقا، اعطای پژوهانه و پروژه‌های تحقیقاتی استفاده می‌شود. علاوه بر موارد فوق، از ارجاعات به عنوان ابزاری کمکی در فرایندهای بازیابی اطلاعات مانند خوشه‌بندی خودکار مدارک و نمایه‌سازی نیز استفاده می‌شود. استفاده از ارجاعات با اهداف ذکر شده نیازمند ساخت شبکه تحلیل استنادی است تا روابط بین موجودیت‌هایی مانند اسناد، پژوهشگران، نشریات و ... مشخص گردد.

برای ساخت شبکه تحلیل استنادی بر روی مجموعه‌ای از اسناد علمی، مجموعه‌ای از اقدامات باید صورت گیرد: ۱- فراداده‌های متناظر با سند ورودی استخراج شود. ۲- سپس، بخش مراجع هر سند استخراج شده و این بخش به مجموعه‌ای مجزا از رشته‌های مرجع تقسیم شود. ۳- در ادامه، باید فراداده‌های متناظر با هر رشته مرجع مانند اسامی پژوهشگران، عنوان پژوهش، محل انتشار پژوهش، سال و ... استخراج شود. ۴- در نهایت باید از الگوریتم‌های تطبیق رشته برای بررسی مطابقت رشته‌های مرجع و مولفه‌های آن‌ها استفاده گردد.

انجام مراحل فوق توسط کاربر انسانی به راحتی قابل انجام است، اما ماشینی‌سازی هر یک از این مراحل دارای چالش‌هایی اساسی است.

استخراج فراداده‌های موردنیاز از سند ورودی به معنای مشخص کردن، نام نویسنده(گان)، عنوان، محل نشر، سال نشر و ... سند ورودی است. با توجه به گستردگی موضوع و همچنین وجود این داده‌ها برای بخش قابل توجهی از اسناد علمی موجود در پایگاه اطلاعات علمی ایرانیان (گنج)، در این پژوهش به این مساله پرداخته نخواهد شد.

مساله دوم یعنی استخراج بخش مراجع یک سند نیز علی‌رغم ظاهر ساده، همواره به آسانی قابل انجام نیست چرا که تعیین قاطع مجموعه کلیدواژه‌های به کار رفته برای آغاز این بخش ممکن نیست. علاوه‌براین، تعیین نقطه پایان این بخش با چالش بیشتری روبرو است چرا که ممکن است بخش مراجع، آخرین بخش یک سند نباشد و برای مثال به طور معمول در اسناد از نوع پارسا، بخش‌های ضمیمه یا لغت‌نامه‌ها پس از بخش مراجع ذکر می‌شوند.

تجزیه رشته‌های مرجع به اقلام اطلاعاتی سازنده با چالش‌های بیشتری مواجه است. به دلایلی مانند اشتباهات صورت گرفته توسط پژوهشگران در وارد کردن داده‌ها، قالب‌های متفاوت به کار رفته در نگارش رشته‌های مرجع و عدم وجود استانداردی یکتا بدین منظور، اشکالات موجود در نرم‌افزارهای جمع‌آوری مراجع، به کار بردن اختصارات در نوشتن اسامی و محل‌های نشر، حذف برخی از قسمت‌ها در نگارش نام نویسندگان و محل‌های نشر چندبخشی و حجم زیاد داده‌ها، خودکارسازی این مسأله به سادگی امکان‌پذیر نیست. تجزیه مولفه نویسندگان یک رشته مرجع به اسامی نویسندگان مجزا، مساله‌ای است که پیچیدگی‌های موجود در مساله تجزیه رشته‌های مرجع را افزایش می‌دهد. دلیل این امر این است که تشخیص نام از نام خانوادگی و همچنین مرز اسامی چندبخشی کاری دشوار است. جزئیات بیشتر درباره چالش‌های موجود در مساله تجزیه رشته‌های مرجع را می‌توان در (پاک‌نیت و همکاران، ۱۳۹۸) یافت.

اشتباهات صورت گرفته توسط پژوهشگران در وارد کردن داده‌ها، اشکالات موجود در نرم‌افزارهای جمع‌آوری مراجع، به کار بردن اختصارات در نوشتن اسامی و محل‌های نشر، حذف برخی از قسمت‌ها در نگارش نام نویسندگان و محل‌های نشر چند بخشی و خطاهای پیش آمده در تجزیه رشته‌های مرجع، خودکارسازی بررسی انطباق رشته‌های مرجع و مولفه‌های مختلف آن‌ها را دشوار نموده است.

روش‌ها و سامانه‌های زیادی برای ساخت شبکه تحلیل استنادی با ورودی یک پایگاه از اسناد به زبان انگلیسی موجود است که هر یک به نحوی قابل‌قبول گام‌های ذکر شده در بالا را انجام داده و شبکه استنادی مربوطه را ایجاد می‌کنند؛ اما از یک طرف، تاکنون روشی بدین منظور برای زبان فارسی ارائه نشده است و از طرف دیگر، با توجه به وابستگی دو گام اول ذکر شده برای ساخت شبکه تحلیل استنادی به زبان، امکان استفاده از روش‌ها و سامانه‌های موجود برای ساخت شبکه تحلیل استنادی روی مجموعه‌ای از اسناد به زبان فارسی وجود ندارد.

با توجه به نقش پژوهشگاه علوم و فناوری اطلاعات ایران (ایرانداک) در نگهداری، نمایه‌سازی و ارائه دانش افزوده از مدارک علمی کشور (برای مثال با هدف ارزیابی پژوهش‌ها و پژوهشگران و بازیابی پارساهای مرتبط)، در این طرح پژوهشی به مسأله ساخت خودکار شبکه تحلیل استنادی بر روی مجموعه پارساهای موجود در پایگاه (گنج) می‌پردازیم. در این راستا، با در نظر گرفتن روش‌های موجود برای ساخت خودکار شبکه تحلیل استنادی در زبان انگلیسی و همچنین ویژگی‌های زبان فارسی و مراجع در این زبان، الگوریتم‌های موردنیاز برای ساخت خودکار شبکه تحلیل استنادی ارائه خواهد شد. در ادامه، روش ارائه شده پیاده‌سازی شده و کیفیت کل آن و همچنین کیفیت بخش‌های مختلف آن بررسی خواهد شد.

الگوریتم‌های پیشنهادی در روش ارائه شده مبتنی بر روش یادگیری ماشین بردار پشتیبان بوده که از روش‌های یادگیری ماشینی با نظارت است. با توجه به این مساله و نیاز این روش‌ها به مجموعه داده‌هایی برچسب‌گذاری شده برای آموزش و آزمایش و همچنین عدم وجود چنین مجموعه داده‌هایی، در این پژوهش و به عنوان بخشی دیگر از مشارکت آن، مجموعه داده‌های موردنظر با استفاده از مجموعه‌ای از پارساهای موجود در پایگاه (گنج) ایجاد شده و در پیاده‌سازی و ارزیابی روش پیشنهادی مورد استفاده قرار خواهد گرفت.

در خاتمه، پیاده‌سازی نهایی روش پیشنهادی با توجه به معیارهای مشخص شده صورت گرفته و حاصل، تحت عنوان یک ماژول نرم‌افزاری برای افزوده شدن به پایگاه گنج خروجی داده می‌شود.

۱-۲- اهداف طرح

هدف این طرح پژوهشی عبارت است از ارائه یک روش خودکار در راستای ایجاد شبکه استنادی پارساهای موجود در پایگاه گنج و در ادامه پیاده‌سازی روش پیشنهادی و ارائه آن به عنوان یک ماژول نرم‌افزاری. این هدف اصلی را می‌توان به اهداف جزئی زیر تقسیم‌بندی نمود:

- ارائه روشی هوشمند جهت تشخیص نوع رشته‌های مرجع.
- بهبود روش هوشمند تجزیه رشته‌های مرجع ارائه شده در (پاک‌نیت و همکاران ۱۳۹۷) به طوریکه روش بهبودیافته ۱) قادر به تجزیه همزمان رشته‌های مرجع در زبان‌های فارسی و انگلیسی باشد و ۲) قادر به تجزیه رشته‌های مرجع به مولفه‌های سازنده آن در سطحی ریزتر باشد به طوری که زیرمولفه‌های مولفه مانند نام نویسندگان نیز به دست آید.
- ارائه روشی جهت بررسی تطابق رشته‌های مرجع و مولفه‌های مختلف آن‌ها.

۱-۳- ساختار این طرح پژوهشی

ادامه این طرح پژوهشی به شرح زیر می‌باشد: در فصل دوم به پیشینه پژوهش پرداخته خواهد شد. در فصل سوم، مفاهیم پیش‌نیاز لازم برای درک ادامه این طرح پژوهشی شامل روش ماشین بردار پشتیبان و معیارهای شباهت جاکارد و فاصله ویرایش لوناشتاین مورد مطالعه قرار خواهد گرفت. در ادامه، جزئیات روش پیشنهادی برای ایجاد شبکه استنادی پارساهای موجود در پایگاه گنج و همچنین ارزیابی آن و ارزیابی بخش‌های مختلف آن در فصل چهارم بیان خواهد شد. نتیجه‌گیری و کارهای آینده در فصل پنجم ذکر شده است.

فصل دوم:

پیشینه پژوهش

استخراج فراداده‌ها از متون، تجزیه رشته‌های مرجع به مولفه‌های سازنده و تطبیق رشته‌های مرجع و مولفه‌های آن‌ها مساله‌های اصلی در ساخت شبکه تحلیل استنادی یک پایگاه داده علمی هستند. همانطور که ذکر شد، در این پژوهش تنها به دو مساله آخر، یعنی تجزیه رشته‌های مرجع به مولفه‌های سازنده و تطبیق رشته‌های مرجع و مولفه‌های آن‌ها پرداخته خواهد شد و استخراج فراداده‌ها از متون، به پژوهش‌های آتی موکول خواهد شد. در ادامه این بخش، به طور خلاصه، به بررسی کارهای انجام شده بر روی دو مساله‌ی اصلی این پژوهش خواهیم پرداخت.

۲-۱- تجزیه رشته‌های مرجع به مولفه‌های سازنده

با توجه به ادبیات مسأله، روش‌های ارائه شده برای تجزیه رشته‌های مرجع را می‌توان به دو دسته زیر تقسیم‌بندی کرد:

- روش‌های مبتنی بر قاعده[۱]،
- روش‌های مبتنی بر یادگیری ماشین[۲].

تحقیقات صورت گرفته نشان‌دهنده این است که استفاده از روش‌های مبتنی بر یادگیری ماشین در حالت کلی منجر به نتایج بسیار بهتری شده و روش‌های مبتنی بر قاعده تنها در موقعیت‌هایی که بدانیم رشته‌های مرجع مورد آزمایش از مجموعه‌ای مشخص از قالب ها تبعیت کرده و با استفاده از این قالب‌ها ایجاد شده‌اند، کارا هستند. در ادامه این بخش، روش‌های ارائه شده در هر یک از دو دسته موردنظر را بررسی خواهیم کرد. برای کسب اطلاعات بیشتر در مورد روش‌های مرور شده در این بخش، از خوانندگان علاقمند تقاضا می‌شود به مدارک اصلی متناظر با این روش‌ها یا به مرور جامع انجام شده در (پاک‌نیت و همکاران ۱۳۹۷) مراجعه کنند.

---

[۱] Rule based methods

[۲] Machine learning based methods

۲-۱-۱. روش‌های هوشمند تجزیه رشته‌های مرجع مبتنی بر قاعده

در این بخش، به طور خلاصه، به بررسی روش‌هایی که با استفاده از قواعد (اکتشافی) به تجزیه رشته‌های مرجع می‌پردازند، خواهیم پرداخت.

در (Besagni et al., 2003)، نویسندگان با انجام تحلیل‌هایی ساختاری و نحوی یک روش تجزیه رشته‌های مرجع در مقالات اسکن شده ارائه کرده‌اند. در (Ding et al., 1999)، نویسندگان با استفاده از مجموعه‌ای از قواعد به تجزیه رشته‌های مرجع در مقالات چاپی و برخط پرداخته‌اند. نتایج به دست آمده نشانگر نتایج بهتر در مورد مقالات چاپی به دلیل رعایت کردن بهتر فرمت‌های نگارش مراجع بوده است. در (Huang et al., 2004)، پایگاه داده‌ای شامل قالب‌های شناخته شده از رشته‌های مرجع ایجاد شده و سپس تجزیه رشته‌های مرجع با استفاده از این پایگاه داده و یکی از روش‌های هم‌ترازسازی توالی به نام BLAST انجام می‌شود. در (Gupta et al., 2009)، از عبارات منظم و همچنین اطلاعاتی پایه‌ای در زمینه موردنظر (مانند لیستی از عناوین نشریات برای برچسب‌زنی به یک قسمت به عنوان نشریه) استفاده شده و روشی دیگر برای تجزیه رشته‌های مرجع ارائه شده است.

۲-۱-۲. روش‌های هوشمند تجزیه رشته‌های مرجع مبتنی بر یادگیری ماشین

مسأله تجزیه رشته‌های مرجع را می‌توان به عنوان یک مسأله "برچسب‌گذاری دنباله‌ای"[۱] یا یک مسأله دسته‌بندی چند دسته‌ای در نظر گرفت. با توجه به کارایی مناسب روش‌های یادگیری ماشین در حل مسائل برچسب‌گذاری دنباله‌ای و دسته‌بندی چند دسته‌ای، در سالیان گذشته به طور گسترده از روش‌های یادگیری ماشین برای حل مسأله تجزیه رشته‌های مرجع نیز استفاده شده است. در (Hetzner, 2008)، از مدل مخفی مارکوف (HMM) برای تجزیه رشته‌های مرجع استفاده شده است. HMM استفاده شده در این مقاله فراوانی لغات و اطلاعات مکانی لغات در اقلام اطلاعاتی را در نظر می‌گیرد. با این حال، این مدل ترتیب حضور لغات در اقلام اطلاعاتی را در نظر نمی‌گیرد. برای در نظر گرفتن این مهم و بهبود نتایج، در (Yin et al., 2004)، از HMM دوتایی برای حل مساله تجزیه رشته‌های مرجع استفاده شده است که روابط دنباله‌های دوتایی لغات را نیز در نظر می‌گیرد. در (Ojokoh et al., 2011)، نویسندگان از مدل HMM سه‌تایی برای حل مسأله تجزیه رشته‌های مرجع استفاده کرده‌اند. در روش جدید از یک ماتریس انتقال سه‌بعدی استفاده شده که در نتیجه آن، انتقال به یک حالت نه تنها به حالت قبل، بلکه به حالت ماقبل‌تر نیز وابسته

---

[۱] Sequential labeling

است. در برخی دیگر از روش‌های موجود، مسأله تجزیه رشته‌های مرجع به عنوان یک مسأله برچسب‌زنی دنباله‌ای در نظر گرفته شده و از میدان‌های تصادفی شرطی (CRF) برای حل آن استفاده شده است. با توجه به بررسی‌های صورت گرفته، CRF استفاده شده در کارهای انجام گرفته در این راستا مشابه بوده و تفاوت کارهای انجام گرفته در این زمینه در ویژگی های به کار رفته در آن‌ها و همچنین زمینه علمی که در آن مسأله تجزیه رشته‌های مرجع مورد بررسی قرار گرفته، است (Councill et al., 2008)، (Lopez, 2009)، (Zou et al., 2010)، (Zhang et al., 2011)، (Kim et al., 2012)، (Peng and McCallum, 2013)، (Tkaczyk et al., 2014) و (Tkaczyk et al., 2015).

در (Zou et al., 2010)، نویسندگان مسأله تجزیه رشته‌های مرجع را به عنوان یک مسأله دسته‌بند چند دسته‌ای در نظر گرفته و قلم اطلاعاتی متناظر با هر لغت موجود در رشته مرجع را با استفاده از دسته‌بند چنددسته‌ای SVM مشخص می‌کنند. علاوه‌براین، نویسندگان توجه کرده‌اند که قواعدی اکتشافی وجود دارند که همیشه در مورد رشته‌های مرجع صحیح هستند. با توجه به این مهم، نویسندگان پس از دسته‌بندی هر لغت موجود در دنباله لغات یک رشته مرجع مورد آزمایش، از این قواعد اکتشافی استفاده کرده و وجود تناقض با این قواعد را بررسی می‌کنند. برای طراحی دسته‌بند موردنظر، در (Zou et al., 2010) از ۱۵ ویژگی استفاده شده است: ۳ ویژگی برای بررسی وجود لغت در لغت‌نامه‌هایی از اسامی نویسندگان، عنوان مقالات و عنوان نشریات (ایجاد شده از داده‌های ۱۰ سال MEDLINE)، یک ویژگی برای بررسی مشابهت لغت با فرمت شماره صفحه، یک ویژگی برای بررسی شباهت لغت با فرمت نگارش نام یک نویسنده، (مانند N. و H.-N.)، یک ویژگی برای بررسی عددی ۴ رقمی و کمتر از سال فعلی بودن لغت، یک ویژگی برای بررسی بودن لغت به صورت‌های et، al، et. یا al. ، یک ویژگی برای بررسی بودن لغت به صورت مشخص‌کننده شماره صفحه (p.، pp. ،p یا pp)، یک ویژگی برای بررسی خاتمه یافتن لغت با "."، یک ویژگی برای بررسی بزرگ یا کوچک بودن حرف اول لغت، یک ویژگی برای بررسی وجود کاراکتری غیر از حرف در لغت، یک ویژگی برای بررسی وجود تنها رقم در لغت، یک ویژگی برای بررسی وجود همزمان عدد و حرف در لغت، یک ویژگی برای بررسی وجود تنها عدد و حرف در لغت و یک ویژگی برای نمایش موقعیت نرمال شده لغت در رشته مرجع (نرمال شده با توجه به تعداد لغات موجود در رشته مرجع). علاوه‌براین، از آن‌جا که در مسأله تجزیه رشته‌های مرجع، لغات همسایه با احتمال بالاتری برچسب یکسانی خواهند داشت، در اینجا، در دسته‌بندی هر لغت، ویژگی‌های لغات قبل و بعد آن نیز در نظر گرفته می‌شوند. در (Zhang et al., 2011)،

نویسندگان بیان کرده‌اند که با توجه به ساختار منظم موجود در رشته‌های مرجع، مسأله تجزیه مرجع را می‌توان به عنوان یک مسأله یادگیری دنباله‌ای در نظر گرفت و در ادامه از SVM ساختاری برای حل این مسأله استفاده کردند. ویژگی‌های در نظر گرفته شده برای هر واحد نیز همانند (Zou et al., 2010) در نظر گرفته شده است.

روش‌های بررسی شده در بالا مختص زبان انگلیسی بوده و همانطور که در (پاک‌نیت و همکاران، ۱۳۹۷) نشان داده شده، استفاده از این روش‌ها در زبان فارسی نتایجی کاملا نادرست را در پی خواهد داشت. بررسی پیشینه مساله بیانگر این است که (پاک‌نیت و همکاران ۱۳۹۷) تنها پژوهش انجام شده در زمینه تجزیه رشته‌های مرجع در زبان فارسی است که در آن مساله تجزیه رشته‌های مرجع به عنوان یک مساله دسته‌بندی چندکلاسه در نظر گرفته شده و از دسته‌بند SVM برای حل آن استفاده شده است. جزئیات این روش در فصل چهار موردبازبینی قرار خواهد گرفت.

۲-۲- تطبیق رشته‌های مرجع

با توجه به دلایلی من جمله قالب‌های مختلف در نگارش مراجع، اشتباهات کامپیوتری در پردازش داده‌ها، اشتباهات کاربری و ...، بررسی تطابقت دقیق رشته‌های مرجع نتایج مناسبی ارائه نخواهد کرد. با توجه به این مساله، محققان استفاده از رویکردهای تطبیق تقریبی را برای این هدف پیشنهاد کرده‌اند. در ادامه، مهم‌ترین کارهای ارائه شده در این زمینه را بررسی می‌کنیم.

در (Lawrence et al., 1999a) و (Lawrence et al., 1999b)، استفاده از معیارهای فاصله ویرایش[۱] مانند فاصله لون‌اشتاین[۲] یا لایک‌ایت[۳] یا استفاده از معیارهای فراوانی کلمات[۴] مانند TF-IDF[۵] برای بررسی مطابقت رشته‌های مرجع پیشنهاد شده است. علاوه‌براین، در این مقاله بیان شده که استفاده از اقلام اطلاعاتی[۶] مانند عنوان، سال انتشار و ... منجر به دستیابی به نتایجی بهتر خواهد شد.

---

۱ Edit distance measures

۲ Levenshtein

۳ LikeIt

۴ Word frequency measures

۵ Term Frequency–Inverse Document Frequency

۶ Field

در (Wang et al., 2015)، نویسندگان با استفاده از SimHash[۱]، یک روش جدید برای حل مساله تطبیق رشته‌های مرجع ارائه کرده‌اند. در این روش، در ابتدا رشته‌های مرجع ورودی تجزیه شده و مولفه‌های مختلف آن‌ها به دست می‌آید. در ادامه و پس از انجام برخی پیش‌پردازش‌ها، از SimHash برای بررسی مطابقت مولفه‌های مختلف دو رشته مرجع استفاده شده و با توجه به مجموع مقایسات مابین مولفه‌های مختلف دو رشته مرجع، در مورد تطابق آن‌ها تصمیم‌گیری می‌شود.

در (Pasula et al., 2003)، نویسندگان مساله تطبیق رشته‌های مرجع را به عنوان یک مساله عدم قطعیت شناسه در نظر گرفته و از مدل‌های رابطه‌ای احتمالاتی برای حل این مساله استفاده کرده‌اند.

در (Prasad Rao, 2010)، نویسندگان به مساله تطبیق رشته‌های مرجع در زبان سانسکریت[۲] پرداخته و از الگوریتم Smith-Waterman-Gotoh[۳] به عنوان معیاری برای بررسی شباهت رشته‌های مرجع استفاده کرده‌اند.

در (Clegg and Pledge, 2008)، نویسندگان از معیار فاصله ویرایش لون‌اشتاین برای بررسی تطابق دو رشته مرجع استفاده کرده‌اند. در این راستا، نویسندگان پیشنهاد کرده‌اند که برای بررسی تطابق دو رشته مرجع: ۱-شباهت قلم اطلاعاتی عنوان دو رشته با یکدیگر مقایسه شود، ۲-در صورت به اندازه کافی شبیه بودن عناوین، نویسنده اول دو رشته با یکدیگر مورد مقایسه قرار گیرند و ۳- در صورت به اندازه کافی شبیه بودن نویسنده اول دو رشته مرجع، تطابق و در غیر این صورت عدم تطابق خروجی داده شود.

۲-۳- جمع‌بندی

در این فصل، به بررسی کارهای مرتبط صورت گرفته در راستای دو مساله اساسی و مهم این پژوهش، یعنی تجزیه رشته‌های مرجع و تطبیق رشته‌های مرجع پرداخته شد. همانطور که بیان شد، روش‌های موجود برای تجزیه رشته‌های مرجع را می‌توان به دو دسته روش‌های مبتنی بر قاعده و روش‌های مبتنی بر یادگیری ماشینی تقسیم‌بندی نمود که در این بین، نتایج استفاده از روش‌های مبتنی بر یادگیری ماشینی در مقایسه با نوع دیگر بسیار بهتر است. علاوه‌براین،

---

[۱] SimHash یک تکنیک برای بررسی میزان شباهت دو مجموعه است.

[۲] Sanskrit

[۳] الگوریتم Smith-Waterman-Gotoh از الگوریتم‌های هم‌ترازسازی توالی است و به طور کلی در بیوانفورماتیک مورد استفاده قرار می‌گیرد.

بررسی‌های انجام شده نشان داد که تاکنون سه دسته از روش‌های مبتنی بر یادگیری ماشینی برای حل این مساله استفاده شده که استفاده از روش‌های میدان‌های تصادفی شرطی (CRF) و ماشین بردار پشتیبان (SVM) نتایج تقریبا یکسان و بهتری را در مقایسه با روش‌های مدل مخفی مارکوف (HMM) ارائه داده‌اند.

علاوه‌براین، مرور پیشینه نشان داد که رویکرد در نظر گرفته شده برای بررسی تطابق رشته‌های مرجع یک رویکرد تطبیق تقریبی است. در این رویکرد، از الگوریتم‌های تطبیق رشته تقریبی برای بررسی تطابق مولفه‌های مختلف رشته‌های مرجع استفاده شده و در ادامه از برخی معیارها و الگوریتم‌ها برای بررسی تطابق رشته‌های مرجع استفاده می‌شود.

فصل سوم:

پیش‌نیازها

در این فصل، پیش‌نیازهای لازم برای درک ادامه این پژوهش ارائه خواهد شد. در این راستا، در ابتدا به بررسی روش‌های دسته‌بندی و معیارهای ارزیابی آن‌ها خواهیم پرداخت. سپس، روش دسته‌بندی ماشین بردار پشتیبان (SVM) که ابزار اصلی مورد استفاده در این پژوهش است، مورد بررسی قرار خواهد گرفت. در خاتمه، به الگوریتم‌های تطبیق رشته تقریبی مورد استفاده در این پژوهش پرداخته خواهد شد. لازم به ذکر است که مطالب ارائه شده در بخش‌های ۳-۱ و ۳-۲ برگرفته از (نصیری ۱۳۹۰)، (کارگر ۱۳۹۰)، (نصیری ۱۳۹۶) و (پاک‌نیت و همکاران ۱۳۹۷) هستند.

۳-۱- روش‌های دسته‌بندی و معیارهای ارزیابی آن‌ها

دسته‌بندی به فرایند نسبت دادن یک شی یا نمونه ناشناخته به یک رده یا دسته بر اساس ویژگی‌های آن گفته می‌شود. برای این منظور دسته‌بندی‌کننده یک مدل از داده‌های آموزشی در مرحله آموزش ایجاد کرده و با استفاده از این مدل داده‌های جدید را در مرحله آزمون مورد آزمایش قرار می‌دهد.

در اغلب مسائل دسته‌بندی، برای ارزیابی روش‌ها از پارامترهای دقت[۱]، فراخوانی[۲] و F1 استفاده می‌شود که در ادامه آن‌ها را تعریف خواهیم کرد:

پارامتر دقت برای هر دسته به صورت $\frac{tp}{tp+fp}$ محاسبه می‌شود که tp و fp به ترتیب برابر با تعداد نمونه‌هایی است که به درستی و اشتباها در دسته موردنظر قرار گرفته‌اند. پارامتر فراخوانی برای هر دسته به صورت $\frac{tp}{tp+fn}$ محاسبه می‌شود که tp برابر با تعداد نمونه‌هایی است که به درستی در دسته موردنظر قرار گرفته و fn تعداد نمونه‌هایی است که درحالیکه

[۱] Precision

[۲] Recall

باید در این دسته قرار می‌گرفتند، در دسته دیگری قرار گرفته‌اند. پارامتر F1 نیز برابر با میانگین دو پارامتر دقت و فراخوانی در نظر گرفته می‌شود.

۳-۲- دسته‌بند ماشین بردار پشتیبان (SVM)[۱]

روش ماشین بردار پشتیبان (SVM) از کارامدترین روش‌های دسته‌بندیست که در سال ۱۹۹۵ معرفی شده است (Vapnik, 1995). در این روش، مسأله همواره به صورت یک مساله برنامه‌ریزی ریاضی مدل‌سازی شده و با حل این مدل، جواب تعیین می‌گردد. هدف نهایی در این روش یافتن یک (چند) ابرصفحه برای جداسازی داده‌ها از یکدیگر است. لازم به ذکر است که در برخی موارد ممکن است جداسازی داده‌ها با ابرصفحه‌ها از یکدیگر میسر نباشد و استفاده از ابرصفحه‌ها برای جداسازی منجر به خطا در دسته‌بندی شود. برای حل این معضل می‌توان از منحنی‌ها و یا توابع غیرخطی برای جداسازی استفاده کرد. پس از تعیین ابرصفحه‌های موردنیاز برای جداسازی دسته‌ها از آن‌ها به عنوان معیاری برای تعیین وضعیت داده‌هایی که وضعیت آن‌ها نامشخص است استفاده خواهد شد. در این حالت، هر ابرصفحه‌ای برای دسته‌بندی مناسب نبوده و در صورتی که دقت حاصل از مقداری قابل قبول بیشتر باشد از آن ابرصفحه برای جداسازی استفاده می‌شود. در غیر این صورت باید برای یافتن یک ابرصفحه دیگر با دقت بالاتر تلاش شود.

۳-۲-۱. روش SVM کلاسیک برای داده‌های دو دسته‌ای جدایی‌پذیر خطی

ساده‌ترین نوع روش SVM برای دسته‌بندی داده‌های دو دسته‌ای جدایی‌پذیر خطی روش کلاسیک SVM است که مبنای بسیاری از سایر انواع روش‌های SVM نیز به شمار می‌رود (Vapnik 1995، Vapnik 1998). فرض کنید تعداد Mداده عددی به صورت بردارهای ستونی m بعدی x_1 ،...، x_M متعلق به دسته‌های ۱ یا ۲ در اختیار باشند. متناظر با هر x_i می‌توان یک اسکالر y_i در نظر گرفت به طوری که اگر x_i متعلق به دسته ۱ باشد y_i مقدار ۱ و چنانچه متعلق به دسته ۲ باشد مقدار 1- دریافت کند. با توجه به این دسته‌بندی برای مجموعه داده‌ها، می‌توان نمایشی به صورت $s = \{(x_i, y_i)\}_{i=1}^{M}$ برای آن‌ها در نظر گرفت. در ادامه، در روش SVM هدف این است که مجموعه داده‌های $s = \{(x_i, y_i)\}_{i=1}^{M}$ به وسیله یک ابرصفحه خطی $\bar{w}^T x + \bar{b} = 0$ که $\bar{w}^T \in R^M$ و $\bar{b} \in R$ به نحوی از یکدیگر جدا شوند که رابطه زیر برقرار باشد:

---

[۱] Support Vector Machine

$$\begin{cases} \bar{w}x_i + \bar{b} > 0 & y_i = 1 \\ \bar{w}x_i + \bar{b} < 0 & y_i = -1 \end{cases} \quad i = 1,2,,\dots,M. \qquad (۱-۳)$$

رابطه فوق را می‌توان به صورت زیر بازنویسی کرد:

$$\exists \varepsilon > 0 \; s.t.: \begin{cases} \bar{w}x_i + \bar{b} \geq \varepsilon & y_i = 1 \\ \bar{w}x_i + \bar{b} \leq \varepsilon & y_i = -1 \end{cases} \quad i = 1,2,,\dots,M \qquad (۲-۳)$$

با ضرب طرفین نامعادلات موجود در رابطه فوق در عدد $\frac{1}{\varepsilon} > 0$ و جایگذاری $w = \frac{1}{\varepsilon}\bar{w}$ و $b = \frac{1}{\varepsilon}\bar{b}$ خواهیم داشت:

$$\begin{cases} w^T x_i + b \geq 1 & y_i = 1 \\ w^T x_i + b \leq -1 & y_i = -1 \end{cases} \quad i = 1,2,,\dots,M. \qquad (۳-۳)$$

یا به عبارت دیگر

$$y_i(w^T x_i + b) \geq 1 \qquad i = 1,2,,\dots,M \qquad (۴-۳)$$

تعریف. مجموعه داده‌های $s = \{(x_i, y_i)\}_{i=1}^{M}$ را جداپذیر خطی[۱] گویند هرگاه $w^T \in R^m$ و $b \in R$ موجود باشند به‌طوریکه با استفاده از این پارامترها رابطه فوق برای مجموعه داده s برقرار باشد. در این صورت، ابرصفحه $w^T x + b = 0$، ابرصفحه جداکننده داده‌های s نامیده می‌شود. علاوه بر این، دو ابرصفحه $w^T x + b = 1$ و $w^T x + b = -1$ ابرصفحات حاشیه‌ای[۲]، ناحیه بین این دو ابر صفحه حاشیه سخت[۳]، فاصله بین این دو ابرصفحه پهنای حاشیه سخت و بردار x_i که در یکی از ابرصفحات حاشیه‌ای صدق کند بردار پشتیبان[۴] نامیده می‌شوند.

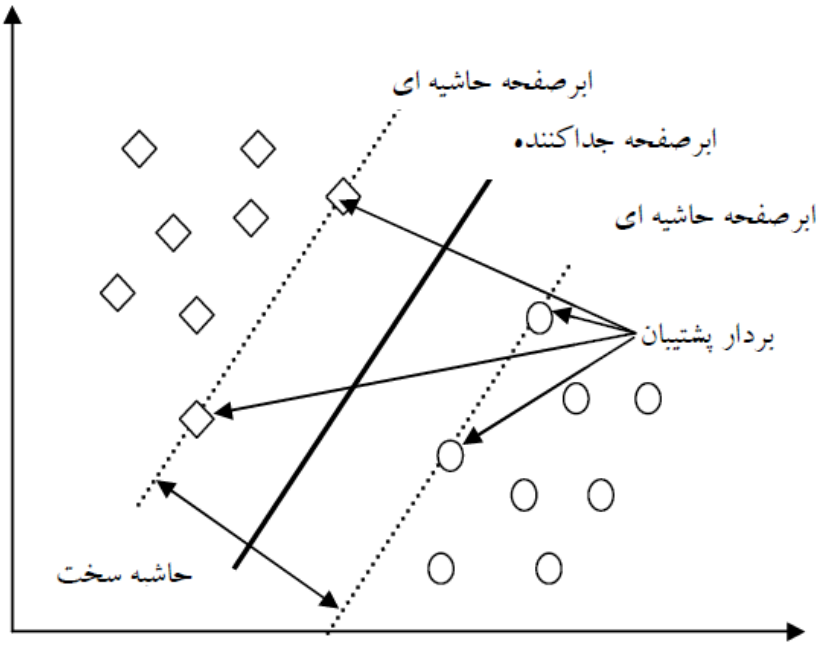
---

[۱] Linearly Seperable

[۲] Marginal Hyperplane

[۳] Hard Margin

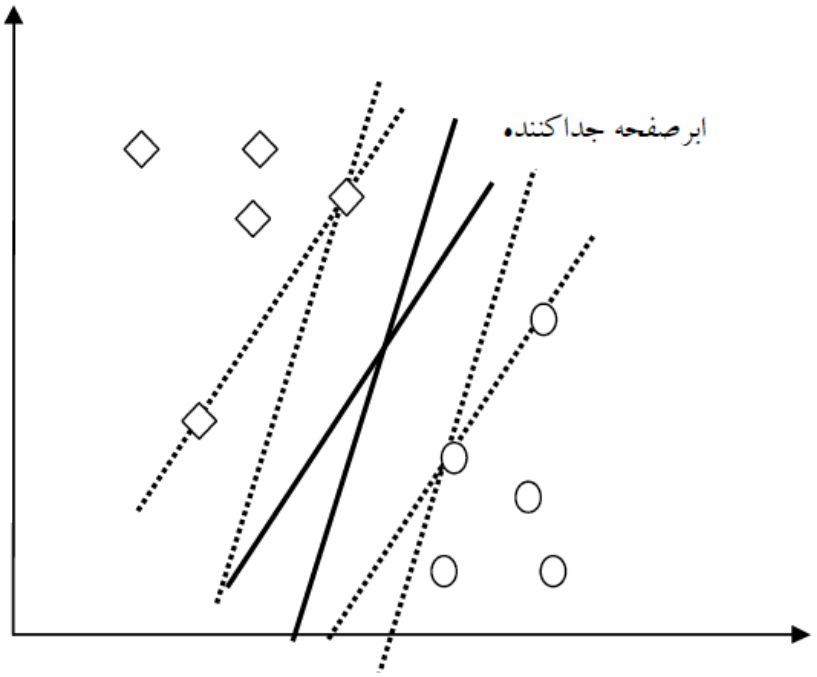
[۴] Support Vector



شکل ۱ . نمایش ابرصفحات حاشیه‌ای و ابرصفحه جداکننده برای تعدادی داده (کارگر ۱۳۹۰)

در شکل ۱ مثالی از ابرصفحات حاشیه‌ای، ابرصفحه جداکننده، حاشیه سخت و بردار پشتیبان برای مجموعه‌ای فرضی از داده‌ها در فضای دو بعدی نشان داده شده است.

همانطور که در شکل ۲ نیز نشان داده شده است برای هر مجموعه داده جداپذیر خطی ابرصفحه‌های جدا کننده زیادی می‌توان پیدا کرد. پهنای حاشیه سخت برخی از این ابرصفحه‌ها کوچک و برخی دیگر بزرگ‌تر است. در روش SVM ترجیح بر این است که ابرصفحه جداکننده به نحوی محاسبه شود که پهنای حاشیه سخت متناظر بیشترین مقدار ممکن را داشته باشد.



شکل ۲. وجود ابرصفحات جداکننده متعدد برای یک مجموعه داده دو دسته‌ای (کارگر ۱۳۹۰)

با توجه به این‌که ابرصفحات حاشیه‌ای با یکدیگر موازی هستند، بیشترین پهنای حاشیه سخت مربوط به ابرصفحات حاشیه‌ایست که بیشترین فاصله را از یکدیگر داشته باشند. با در نظر گرفتن $\frac{2}{\|w\|}$ به عنوان فاصله بین دو ابرصفحه حاشیه‌ای، هدف در استفاده از SVM عبارت است از کمینه کردن $\frac{1}{2}\|w\|^2$ به‌طوریکه

$$y_i(w^T x_i + b) \geq 1 \qquad i = 1,2,,\dots,M, w \in R^m, b \in R \qquad (۵-۳)$$

تعریف. ابرصفحه جداکننده با بیشترین حاشیه سخت مربوط به مجموعه داده‌های $s = \{(x_i, y_i)\}_{i=1}^{M}$ را ابرصفحه جداکننده بهینه[۱] یا به اختصار ابرصفحه بهینه نامند.

تعریف. تابع $D(x) = w^T x + b$ را تابع تصمیم[۲] مجموعه داده‌های $s = \{(x_i, y_i)\}_{i=1}^{M}$ گویند هرگاه $w^T x + b = 0$ ابرصفحه بهینه جداکننده این مجموعه باشند.

---

۱ Optimal Sperating Hyperplane

۲ Decision Function

همانطور که قبلا نیز بیان شد، هدف در استفاده از SVM عبارت است از استفاده از داده‌هایی با دسته مشخص برای یافتن مدلی جهت تعیین دسته داده‌هایی که دسته آن‌ها مشخص نیست. تابع تصمیم ذکر شده در تعریف فوق همان مدل مورد بحث است. تشخیص دسته داده‌ها با استفاده از این تابع تصمیم به صورت زیر انجام می‌شود:

$$x \in \begin{cases} 1 \text{ دسته } D(x) > 0 \\ 2 \text{ دسته } D(x) < 0 \end{cases} \qquad (۶-۳)$$

اگر $D(x) = 0$، در این صورت x روی ابرصفحه جداکننده بهینه قرار داشته و مدل حاصل قادر به تشخیص دسته x نخواهد بود. در این حالت ممکن است بتوان با تغییر داده‌های اولیه ساخت مدل، ابرصفحه بهینه دیگری به دست آورد تا تعیین کننده وضعیت دسته x باشد.

روش کلاسیک SVM بررسی شده برای داده‌هایی مناسب است که جداپذیر خطی باشند. اما، در اکثر مسائل واقعی چنین چیزی برقرار نبوده و در نتیجه روش ارائه شده در بخش قبل برای این مسائل بدون استفاده خواهد بود. در ادامه، به بررسی راه‌حل روش SVM برای داده‌هایی که به صورت خطی جداپذیر نیستند، پرداخته خواهد شد.

داده‌هایی که جداپذیر خطی نیستند را می‌توان در یکی از دو حالت زیر در نظر گرفت:

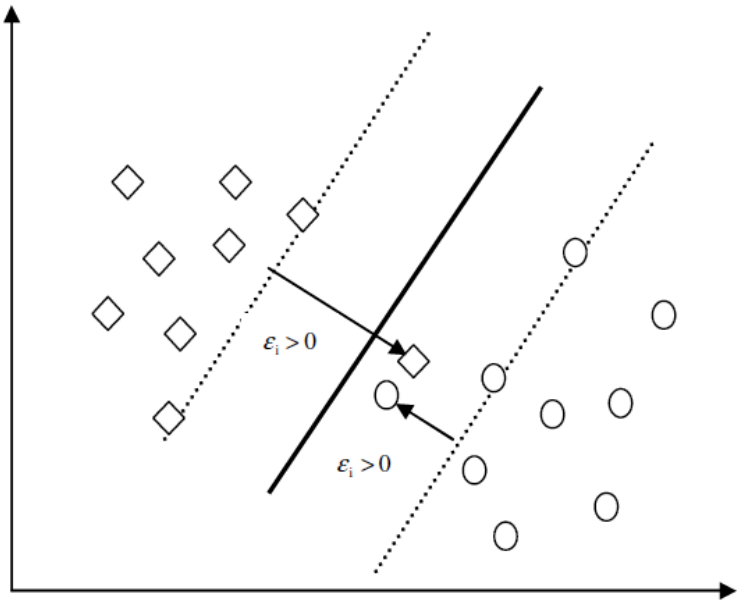
- داده‌هایی که بتوان با قبول درجه‌ای از خطا آن‌ها را با استفاده از یک ابرصفحه جداسازی کرد. این بدین معنی است که ممکن است در این دسته‌بندی دسته اشتباه به تعداد اندکی از داده‌ها اختصاص یابد.
- داده‌هایی که در صورت جداسازی خطی آن‌ها، خطای زیادی حاصل شود به طوری که مدل به دست آمده از این جداسازی فاقد اعتبار باشد. در این حالت ممکن است بتوان جداسازی موردنظر را به صورت غیرخطی (یعنی با استفاده از توابعی غیرخطی) انجام داد. با توجه به عدم استفاده از SVM در این حالت در این پژوهش، جزئیات بیشتر در مورد این حالت بیان نشده و خوانندگان محترم برای اطلاع در مورد این حالت به منابع (Liu 1998، Masulli and Valentini 2000) ارجاع داده می‌شوند.

۳-۲-۲. روش SVM بر اساس نرم‌های $L1$ و $L2$

همانطور که بیان شد جداسازی مجموعه داده‌های دو دسته‌ای $s = \{(x_i, y_i)\}_{i=1}^{M}$ با استفاده از رابطه (۳-۴) اجازه هیچ گونه خطایی در دسته‌بندی نمی‌دهد. برای مجاز کردن خطا رابطه مورد نظر به صورت زیر تغییر داده می‌شود:

$$y_i(w^T x_i + b) \geq 1 - \varepsilon_i \qquad i = 1,2,,\dots,M \quad - \qquad (۳-۷)$$

که ε_i یک متغیر کمبود نامنفی است. شکل ۳ مفهوم ε_i را برای جداسازی داده‌هایی که به صورت خطی جداپذیر نیستند نمایش می‌دهد. در این صورت، اگر $0 < \varepsilon_i < 1$، همانطور که در شکل ۳ نیز دیده می‌شود، داده x_i در درون ناحیه بین دو ابرصفحه حاشیه‌ای قرار گرفته است. اگر $\varepsilon_i \geq 1$، x_i به وسیله ابرصفحه بهینه به دست آمده اشتباها دسته‌بندی شده و در صورتی که $\varepsilon_i = 0$، داده موردنظر به درستی دسته‌بندی شده است. در صورتی که داده اشتباه دسته‌بندی شده باشد مقدار ε_i برابر با فاصله x_i تا ابرصفحه حاشیه‌ای مربوط به دسته درست داده است.



شکل ۳. یک دسته‌بندی خطی همراه با خطا (کارگر ۱۳۹۰)

حال فرض کنید مجموعه داده‌های دو دسته‌ای $s = \{(x_i, y_i)\}_{i=1}^{M}$ به کمک ابرصفحه جداکننده به نحوی دسته‌بندی شوند که برخی از آن‌ها در دسته اشتباه قرار گیرند. در این صورت، در این‌جا برخی از داده‌ها در ناحیه بین دو ابرصفحه حاشیه‌ای قرار خواهند گرفت و ناحیه بین دو ابرصفحه حاشیه‌ای را حاشیه نرم[۱] گویند. در شکل

[۱] Soft marigin

۳ مثالی از یک حاشیه نرم در دسته‌بندی مجموعه‌ای از داده‌های فرضی در یک فضای دو بعدی نشان داده شده است. در این نوع دسته‌بندی، اهداف زیر مدنظر است:

۱- حاشیه نرم متعلق به ابرصفحه جداکننده بهینه بیشترین پهنای ممکن را داشته باشد.
۲- تعداد داده‌های اشتباه دسته‌بندی شده به حداقل ممکن برسد.

هدف اول به دنبال بیشینه کردن فاصله بین دو ابرصفحه حاشیه‌ای است، در حالیکه هدف دوم به دنبال کمینه کردن تعداد داده‌های اشتباها دسته‌بندی شده است. با توجه به عدم وجود سنخیت بین این دو هدف، هدف دوم به عنوان کمینه کردن خطای دسته‌بندی در نظر گرفته می‌شود که خطای دسته‌بندی به صورت زیر تعریف می‌شود:

تعریف: فرض کنید دسته‌بندی مجموعه داده‌های دو دسته‌ای $s = \{(x_i, y_i)\}_{i=1}^{M}$ با استفاده از یک ابرصفحه جداکننده به نحوی انجام شود که رابطه (۳-۷) برقرار باشد. در این صورت، $\|\varepsilon\|_p^p$ به عنوان خطای دسته‌بندی تعریف شده که $p \in N$ و $\varepsilon = (\varepsilon_1, \varepsilon_2, \dots, \varepsilon_M)$.

بسته به اینکه از چه نرمی برای خطای دسته‌بندی داده‌ها در تعریف فوق استفاده شود، حالت‌های مختلفی از روش SVM به دست می‌آید. دو نرم اصلی مورد استفاده عبارتند از $p = 1$ (استفاده از نرم L1) که در این صورت مدل به دست آمده، مدل SVM بر اساس L1 یا به اختصار L1-SVM نامیده شده و $p = 2$ (استفاده از نرم L2) که در این صورت مدل حاصل را مدل SVM بر اساس L2 یا به اختصار L2-SVM گویند.

۳-۲-۳. SVM برای دسته‌بندی داده‌های چند دسته‌ای

در بخش قبل، روش SVM برای دسته‌بندی داده‌های دو دسته‌ای مورد بررسی قرار گرفت. اما، اغلب مسائل در دنیای واقعی، مانند مساله مورد بررسی در این پژوهش، چند دسته‌ای بوده و استفاده از روش‌های دو دسته‌ای به خودی خود برای حل این مسائل مفید نیست. چندین رویکرد برای گسترش روش SVM با هدف استفاده از آن در مسائل چند دسته‌ای ارائه شده که در این بخش، دو نوع متداول‌تر آن‌ها مورد بررسی قرار خواهد گرفت. برای کسب اطلاعات بیشتر در مورد سایر روش‌های دسته‌بندی SVM چند دسته‌ای به (کارگر ۱۳۹۰) مراجعه شود:

- روش SVM چند دسته‌ای با رویکرد یک دسته در مقابل سایر دسته‌ها[۱] که به اختصار OSVM نامیده می‌شود.
- روش SVM چند دسته‌ای با رویکرد یک دسته در مقابل هر دسته[۲] دیگر که به اختصار PSVM نامیده می‌شود.

بنابر فرمول‌بندی ارائه شده توسط وپنیک (Vapnik 1995)، در OSVM تبدیل یک مساله k دسته‌ای به k مساله دو دسته‌ای، از ابتدایی‌ترین روش‌های دسته‌بندی داده‌های چند دسته‌ای بوده از کارایی پایینی برخوردار است. در این روش برخی از نواحی در فضای مساله غیر دسته‌بندی شده باقی خواهد ماند. برای بهبود این مشکل، در روش PSVM ارائه شده توسط کربل (Krebel 1999) یک مساله k دسته‌ای به k(k-1)/2 مساله دو دسته‌ای که شامل همه ترکیبات دو دسته‌ای از مجموع k دسته است، تبدیل می‌شود. در ادامه جزئیات این روش‌ها را مورد بررسی قرار خواهیم داد.

روش OSVM برای دسته‌بندی داده‌های چند دسته‌ای

مجموعه داده‌های $s = \{(x_i, \bar{y}_i)\}_{i=1}^{M}$ که $x_i \in R^m (i = 1,2, \dots, M)$، $\bar{y}_i \in \{1,2, \dots, k\}$، $k > 2$ تعداد دسته‌ها و M تعداد داده‌ها است، را در نظر بگیرید. در روش OSVM، هر بار یکی از دسته‌ها از سایر دسته‌ها جداسازی می‌شود. برای این کار با فرض اینکه جداسازی دسته jام از سایر دسته‌ها مدنظر باشد، داده‌ها به صورت مساله‌ای دو دسته‌ای که در آن دسته اول شامل داده‌های دسته jام (متناظر با $y_i = 1$) و دسته دوم شامل داده‌های سایر دسته‌ها (متناظر با $y_i = -1$) در نظر گرفته می‌شوند. به عبارت دیگر:

$$y_i = \begin{cases} 1 & \bar{y}_i = j \\ -1 & \bar{y}_i \neq j \end{cases} \qquad i = 1, \dots, M \qquad (۸-۳)$$

در ادامه، جداسازی به وسیله روش‌های SVM دو دسته‌ای انجام می‌شود. این عمل برای تمام دسته‌ها، یعنی برای $j = 1,2, \dots, k$، انجام می‌شود.

تعریف. مجموعه داده‌های k دسته‌ای $s = \{(x_i, \bar{y}_i)\}_{i=1}^{M}$ را در نظر بگیرید. فرض کنید برای $j = 1, \dots, k$، در روش OSVM تابع تصمیم $D_j(x) = w_j{}^T x + b_j$ برای جداسازی داده‌های دسته jام از داده‌های سایر دسته‌ها به دست آمده

---

۱ One- Against- all SVM

۲ Pairwise SVM

باشد. در این صورت، در صورت وجود j–ای یکتا که $D_j(x) > 0$، آنگاه داده x متعلق به دسته jام است. در صورتی که رابطه فوق به ازای چند j برقرار باشد و یا این رابطه به ازای هیچ مقداری از j برقرار نباشد، داده x قابل دسته‌بندی نیست.

در روش OSVM ممکن است بسیاری از داده‌ها قابل دسته‌بندی نباشند. دلیل این امر این است که در این روش برخی از نواحی فضای دسته‌بندی در هیچ یک از دسته‌ها قرار نمی‌گیرند. به عبارت دیگر، وجود نواحی دسته‌بندی نشده در این روش از مشکلات آن محسوب می‌شود. برای بهبود این مشکل، راه‌حل‌های متفاوتی ارائه شده که یکی از آن‌ها روش PSVM است که در ادامه به آن پرداخته خواهد شد. لازم به ذکر است که روش SVM نیز مشکل نواحی دسته‌بندی نشده را به طور کامل حل نمی‌کند و راه‌حل‌های دیگری نیز برای این مساله ارائه شده که با توجه به عدم تمرکز این پژوهش بر روی این مساله، در این‌جا به آن‌ها پرداخته نشده و خوانندگان گرامی به (Amari 1999، LeCun and Bottou 1998، Masulli and Valentini 2000، Vapnik 1998 و Bradley and Mangasarian 1998) ارجاع داده می‌شوند.

روش PSVM برای دسته‌بندی داده‌های چند دسته‌ای

در روش PSVM یافتن ابرصفحه جداکننده بهینه برای هر ترکیب دوتایی از k دسته موجود انجام می‌شود. به عبارت دیگر، در این روش k(k-1)/2 دسته‌بندی انجام می‌شود. برای جداسازی دسته i ام از دسته j ام، تنها از داده‌های مربوط به این دو دسته استفاده می‌شود. داده‌های دسته i ام متناظر با $y_t = 1$ و داده‌های مربوط به دسته j ام متناظر با $y_t = -1$ در نظر گرفته می‌شود. به عبارت دیگر برای داده متعلق به یکی از دو دسته i یا j، رابطه زیر تعریف می‌شود:

$$y_t = \begin{cases} 1 & \bar{y}_t = i \\ -1 & \bar{y}_t = j \end{cases} \qquad t = 1, \dots, M \qquad (۹-۳)$$

سپس جداسازی به وسیله یکی از روش‌های SVM انجام می‌شود. این کار برای هر ترکیب دوتایی از k دسته انجام می‌شود.

تعریف. فرض کنید $w_{ij} \in R^m$ و $b_{ij} \in R$ پارامترهای ابرصفحه بهینه جداکننده دو دسته i و j برای مجموعه داده‌های k دسته‌ای $(k > 2)$ $s = \{(x_i, \bar{y}_i)\}_{i=1}^{M}$ باشند که به وسیله یکی از روش‌های دسته‌بندی SVM دو دسته‌ای به دست

آمده‌اند و $x_i \in R^m (i = 1,2, \dots, M)$ و $\bar{y}_i \in \{1,2, \dots, k\}$. در این صورت تابع تصمیم جداسازی دو دسته i و j از یکدیگر به صورت زیر در نظر گرفته می‌شود:

$$D_{ij}(x) = w_{ij}{}^T x + b_{ij} \qquad i, j = 1,2, \dots, k, i \neq j \qquad (۱۰-۳)$$

با توجه به تعریف تابع تصمیم همواره $D_{ij}(x) = -D_{ji}(x)$ خواهد بود.

بنابراین، در روش PSVM برای یک مجموعه داده k دسته‌ای تعداد $k(k-1)/2$ تابع تصمیم به دست آمده که دسته‌ها را دو به دو از یکدیگر جداسازی می‌کند به طوری که اگر $D_{ij}(x) > 0$، داده x نسبت به دسته jام در دسته iام قرار دارد. هدف مساله یافتن دسته یک داده نسبت به کل دسته‌هاست.

تابع تصمیم در روش PSVM به صورت زیر است:

$$D_i(x) = \sum_{i=1, i \neq j}^{k} \operatorname{sgn}(D_{ij}(x)) \qquad (۱۱-۳)$$

که

$$\operatorname{sgn}(z) = \begin{cases} 1 & z \geq 0 \\ -1 & z < 0 \end{cases} \qquad (۱۲-۳)$$

در این صورت اگر r در رابطه زیر دارای جواب یکتا i باشد، x متعلق به دسته i خواهد بود. در غیر این صورت، x با این روش قابل دسته‌بندی نیست:

$$\max_{r=1,2,\dots,k} D_r(x) \qquad (۱۳-۳)$$

۳-۳- الگوریتم‌های تطبیق رشته تقریبی

الگوریتم های متنوعی برای تطبیق تقریبی رشته ارائه شده است. در ادامه، دو الگوریتم مورد استفاده در این پـژوهش، یعنی فاصله ویرایش لونااشتاین و الگوریتم تشابه n-تایی‌ها را بررسی خواهیم کرد.

۳-۳-۱. الگوریتم تشابه n-تایی‌ها

n-تایی‌ها یا n-گرام‌ها عبارتند از رشته‌های فرعی به طول n در رشته‌های طولانی‌تر. n-تایی‌های رایج عبارتند از یونیگرام (n=1)، بیگرام (n=2 هم چنین دیگرام نیز خوانده می‌شوند) و تریگرام‌ها (n=3). برای مثال، کلمه peter دارای دوتایی‌های pe، et و te و er است. با استفاده از n-تایی‌ها، می‌توان معیارهای شباهت متفاوتی را در نظر گرفت و استفاده کرد که از آن جمله می‌توان به موارد زیر اشاره کرد (نصیری ۱۳۹۷):

۱- **ضریب هم‌پوشانی:** فرض کنیم s_1 و s_2 دو مجموعه از n-تایی‌ها باشند. در این صورت، میزان شباهت دو مجموعه s_1 و s_2 با توجه به ضریب هم‌پوشانی به صورت زیر به دست می‌آید:

$$sim(s_1, s_2) = \frac{|s_1 \cap s_2|}{\min(|s_1|,|s_2|)}$$

۲- **معیار ژاکارد**[۱]: فرض کنیم s_1 و s_2 دو مجموعه از n-تایی‌ها باشند. در این صورت، میزان شباهت دو مجموعه s_1 و s_2 با توجه به معیار ژاکارد به صورت زیر به دست می‌آید:

$$sim(s_1, s_2) = \frac{|s_1 \cap s_2|}{|s_2 \cup s_2|}$$

۳- **ضریب دایس:** فرض کنیم s_1 و s_2 دو مجموعه از n-تایی‌ها باشند. در این صورت، میزان شباهت دو مجموعه s_1 و s_2 با توجه به ضریب دایس به صورت زیر به دست می‌آید:

$$sim(s_1, s_2) = \frac{2|s_1 \cap s_2|}{|s_2|+|s_2|}$$

۳-۳-۲. فاصله لون‌اشتاین

فاصله لون‌اشتاین یا فاصله ویرایش در نظریه داده و علوم کامپیوتر متری برای محاسبه میزان تفاوت میان دو رشته است. فاصله لون‌اشتاین بین دو رشته به وسیله کمترین تعداد عملیات مورد نیاز برای تبدیل یک رشته به رشته دیگر معین می‌شود، که عملیات‌های ممکن عبارتند از ورود، حذف و جایگزینی. این معیار به افتخار ولادمیر لون‌اشتاین، که این فاصله را در سال ۱۹۵۶مطرح کرد، نام‌گذاری شده‌است. در ابتدایی‌ترین نوع فاصله لون‌اشتاین، هر ویرایش دارای هزینه ۱ است. با استفاده از الگوریتم برنامه‌نویسی پویا، فاصله (تعداد ویرایش‌ها) بین دو رشته s1 و s2 را می‌توان در زمان $O(|s1| \times |s2|)$ با استفاده از فضای $O(\min|s1| \times |s2|)$ محاسبه کرد. فاصله ممکن است با استفاده از فرمول زیر به یک معیار شباهت تبدیل شود (بین ۰,۰ و ۱,۰):

---

[۱] Jaccard

$$\mathrm{Sim}_{\mathrm{ld}}(s1, s2) = 1.0 - \frac{\mathrm{dist}_{\mathrm{ld}}(s1, s2)}{\max(|s1|, |s2|)}$$

در حالی که distld(s1,s2) تابع فاصله واقعی لون‌اشتاین بوده و مقدار ۰ را در صورتی که رشته‌ها یکسان یا تعداد ویرایش‌ها با متفاوت بودن رشته‌ها مثبت باشد بازمی‌گرداند. فاصله لون‌اشتاین متقارن بوده و همیشه به شکل زیر است:

$$\mathrm{abs}(|s1|-|s2|) \leq \mathrm{distld}(s1,s2) \text{ و } 0 \leq \mathrm{distld}(s1,s2) \leq \max(|s1|,|s2|)$$

ویژگی دوم امکان فیلتر سریع جفت رشته‌هایی که تفاوت طولی زیادی دارند را ممکن می‌سازد.

فاصله اولیه‌ی لون‌اشتاین هزینه‌های متفاوت ویرایشی یا حتی هزینه‌هایی را که به کاراکتر وابسته‌اند را ایجاد می‌کند (مانند ، جابجایی از q به g که ممکن است هزینه کمتری در مقایسه با x به i داشته باشد که احتمالا همان علت شباهت شکل آن‌هاست). در سال‌های اخیر، محققین تکنیک‌هایی را برای پی بردن به هزینه‌های ویرایش اطلاعات آموزشی برای بهبود کیفیت انطباق ارائه کرده‌اند (نصیری ۱۳۹۶).

فصل چهارم:

روش پیشنهادی

در این فصل، جزئیات روش پیشنهادی و ارزیابی مولفه‌های مختلف آن را مورد بررسی قرار خواهیم داد.

ساخت شبکه استنادی پارساها نیازمند ایجاد و نگهداری چندین جدول است. جزئیات این جدول‌ها به شرح زیر هستند:

- **جدول اسناد**: جدولی که سطرهای آن نمایانگر اسناد (موجود یا مجازی[۱]) و ستون‌های آن نمایانگر فراداده‌های متناظر با هر سند است. علاوه‌براین، در این جدول به هر سند یک شناسه اختصاص می‌یابد که در کنار فراداده‌های آن سند ذخیره می‌شود. همچنین، در این جدول با استفاده از اطلاعات موجود، اطلاعات دیگری نیز از پیش محاسبه و نگهداری شده تا کارایی فرایند جستجو و تطبیق در شبکه استنادی را افزایش یابد. در حال حاضر، هر سطر این جدول شامل مولفه‌های اطلاعاتی زیر در نظر گرفته شده است:
  - شناسه یکتای سند
  - زبان
  - نوع
  - عنوان
  - سال
  - شماره: این مولفه تنها در مورد اسناد با نوع نشریه پر خواهد شد و برای سایر انواع اسناد مقدار این مولفه برابر با null خواهد بود.

[۱] در این پژوهش، سند موجود سندی است که تمام‌متن آن در پایگاه داده موجود باشد و سند مجازی سندی است که تنها به صورت یک رشته مرجع در انتهای یک سند موجود مورد استناد قرار گرفته باشد و تمام‌متن آن ناموجود باشد.

- شماره صفحه: این مولفه تنها برای اسناد با نوع نشریه و همایش پر شده و برای سایر انواع اسناد، مقدار این مولفه برابر با null خواهد بود.
- آدرس

- **جدول استنادها**: به ازای هر رابطه استنادی موجود در پایگاه داده، سطری در این جدول شامل شناسه سند استناد شده، شناسه سند استنادکننده وجود دارد.
- **جدول پدیدآوران:** وظیفه این جدول نگهداری نام کامل پدیدآوران است. در این جدول، به ازای هر پدیدآور تشخیص داده شده یک سطر ایجاد می‌شود که شامل مولفه‌های زیر است:
  - شناسه پدیدآور
  - نام پدیدآور
- **جدول محل‌ها:** وظیفه این جدول نگهداری نام کامل محل‌ها شامل نشریات، همایش‌ها، دانشگاه‌ها و انتشارات‌ها است. در این جدول، به ازای هر محل تشخیص داده شده یک سطر ایجاد می‌شود که در آن مولفه‌های زیر نگهداری می‌شود:
  - شناسه محل
  - نام محل
  - نوع محل

برای دستیابی به نتایج بهتر و با توجه به امکان دسترسی به لیست مجلات، همایش‌ها، ناشرین و دانشگاه‌ها، می‌توان قبل از شروع به پردازش پارساهای موجود در پایگاه، این لیست‌ها را وارد جدول محل‌ها کرد.

- **جدول روابط اسناد-نویسندگان:** وظیفه این جدول نگهداری روابط بین نویسندگان و اسناد است. در این جدول، به ازای هر یک از نویسندگان یک سند، یک سطر وجود دارد. مولفه‌های نگهداری شده در این جدول به صورت زیر است:
  - شناسه سند
  - شناسه نویسنده
- **جدول روابط اسناد-محل‌ها:** این جدول روابط بین اسناد و محل انجام یا چاپ اسناد را نگهداری می‌کند. به عنوان مثال، در صورتی که یک سند (مجازی) از نوع نشریه باشد، در به ازای آن در این جدول

یک سطر شامل شناسه سند و شناسه متناظر با نشریه موردنظر ایجاد خواهد شد. مولفه‌های نگهداری شده در این جدول به صورت زیر است:

- شناسه سند
- شناسه محل

٤-١- روش پیشنهادی

برای ساخت شبکه استنادی پایگاه داده گنج پیشنهاد می‌شود که به ازای هر یک از پارساهای موجود در این پایگاه داده، گام‌های زیر که در شکل ۴ نیز نشان داده شده‌اند، صورت گیرد:

۱. به ازای هر پارسای موجود و فراداده‌های متناظر با آن مانند نام نویسنده، عنوان، محل نشر، سال انتشار و ...:
   أ. یک شناسه یکتای ID_P به این پارسا اختصاص می‌یابد.
   ب. با توجه به الگوریتم AuthMatch ارائه شده در بخش ۴-۶-۱، جدول پدیدآوران با هدف یافتن شناسه پدیدآور پارسا جستجو می‌شود. در صورت عدم وجود، یک شناسه یکتا به این پدیدآور اختصاص یافته و همراه با نام پدیدآور به جدول پدیدآوران اضافه می‌شود. در هر دو حالت، شناسه پدیدآور، جایگزین نام او در لیست فراداده‌ها می‌شود.
   ت. با توجه به الگوریتم VenueMatch، ارائه شده در بخش ۴-۶-۲، جدول محل‌ها با هدف یافتن شناسه محل نشر این پارسا جستجو می‌شود. در صورت عدم وجود، یک شناسه یکتا به این محل نشر اختصاص یافته و همراه با نام محل در جدول محل‌ها ذخیره می‌شود. در هر دو حالت، شناسه محل نشر، جایگزین نام آن در لیست فراداده‌ها می‌شود.
   ث. شناسه مربوطه همراه با فراداده‌های متناظر با این پارسا به جدول اسناد اضافه می‌شود.
۲. پیش‌پردازشی شامل یکسان‌سازی نویسه‌ها بر روی متن پارسای ورودی اعمال می‌شود.
۳. با استفاده از الگوریتم ExtCits، ارائه شده در بخش ۴-۲، رشته‌های مرجع استناد شده در پارسای ورودی استخراج می‌شوند.
۴. به ازای هر رشته مرجع استناد شده:
   أ. با استفاده از الگوریتم LangDet (توصیف شده در بخش ۴-۳)، زبان رشته مرجع مشخص می‌شود.

ب. با توجه به زبان رشته مرجع، با استفاده از الگوریتم PerTypeDet یا EngTypeDet (توصیف شده در بخش ۴-۴) نوع رشته مرجع مشخص می‌شود.

ت. با توجه به زبان، با استفاده از الگوریتم PerCitPars یا EngCitPars (توصیف شده در ۴-۵) رشته مرجع به مولفه‌های سازنده آن تجزیه می‌شود.

ث. به ازای هر پدیدآور تشخیص داده شده: با استفاده از الگوریتم AuthMatch در جدول پدیدآوران به جستجوی این پدیدآور پرداخته، در صورت عدم وجود، یک شناسه یکتا به پدیدآور جدید اختصاص یافته و زوج (شناسه و نام پدیدآور جدید) به جدول پدیدآوران اضافه می‌شود. در هر دو صورت، شناسه او جایگزین نام او می‌شود.

ج. با استفاده از الگوریتم VenueMatch، در جدول محل‌ها به جستجوی محل تشخیص داده شده پرداخته، در صورت عدم وجود، یک شناسه یکتا به محل جدید اختصاص یافته و زوج (شناسه و نام محل جدید) به جدول محل‌ها اضافه می‌شود. در هر دو صورت، شناسه محل جایگزین نام محل می‌شود.

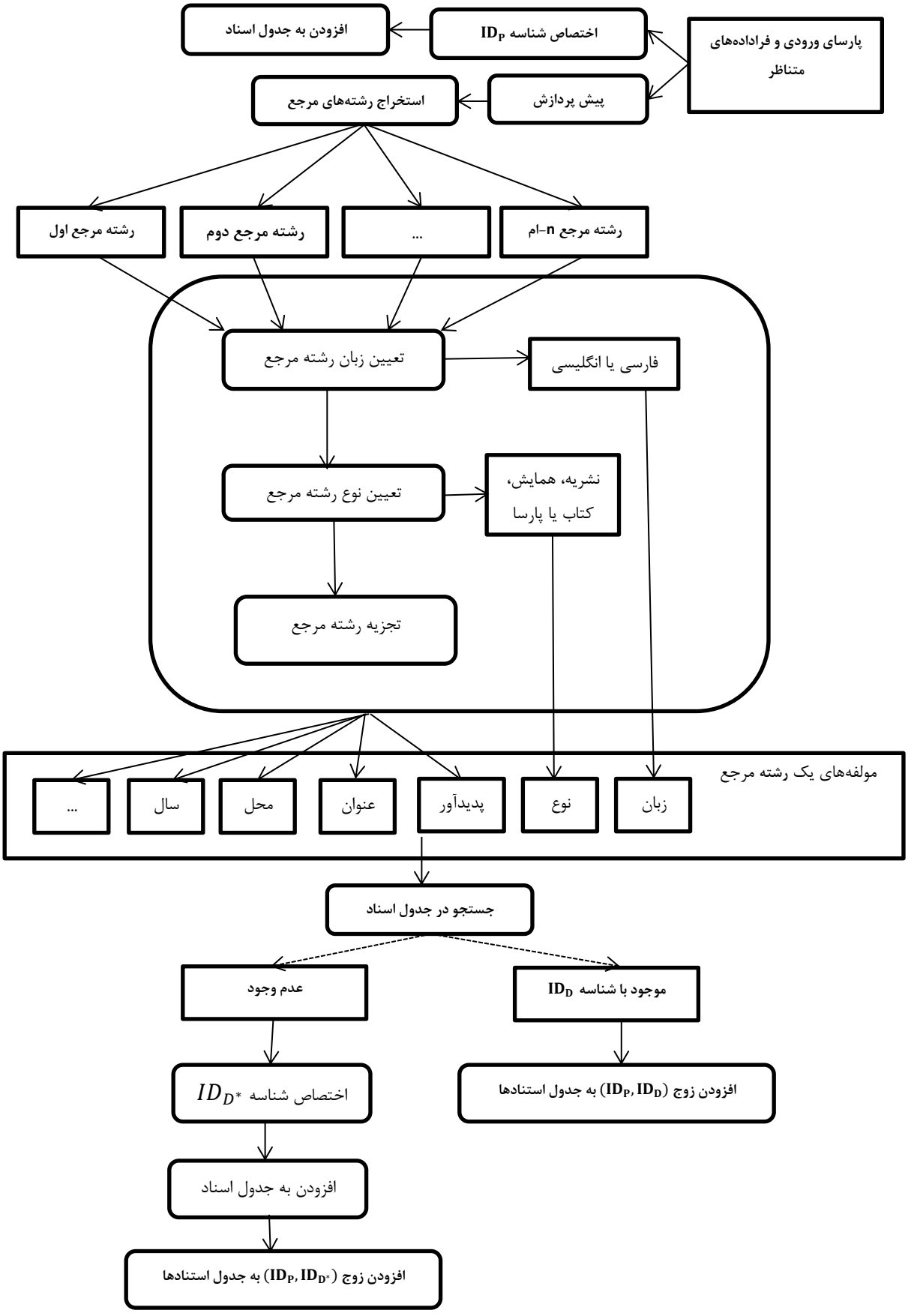
ح. با توجه به مولفه‌های به دست آمده، با استفاده از الگوریتم CitMatch، ارائه شده در بخش ۴-۶-۴، در جدول اسناد به جستجوی سطری پرداخته می‌شود که با این رشته مرجع تطابق تقریبی داشته باشد.

i. در صورت یافتن چنین سطری: فرض کنید، ID_D اندیس سند متناظر با مورد یافته شده باشد. در این صورت زوج جدید (ID_P, ID_D) به جدول استنادها اضافه می‌شود.

ii. در غیر این صورت:

۱. یک شناسه یکتای جدید ID_{D^*} به این سند مجازی اختصاص یافته و همراه با فراداده‌های متناظر با آن به عنوان یک سطر جدید به جدول اسنادی اضافه می‌شود.

۲. زوج جدید $(\mathrm{ID_P}, \mathrm{ID_{D^*}})$ به جدول استنادها اضافه می‌شود.



شکل ٤. مراحل پیشنهادی در راستای ساخت شبکه تحلیل استنادی پایگاه داده گنج.

٤-۲- استخراج رشته‌های مرجع

ممکن است استخراج رشته‌های مرجع از یک پارسا مساله ساده‌ای به نظر رسد: ابتدا به دنبال یافتن کلیدواژه‌های در نظر گرفته شده برای بخش مراجع مثل "منابع" یا "مراجع" گشته و در ادامه، هر پاراگراف ذیل این کلیدواژه تا انتهای پارسا را به عنوان یک رشته مرجع در نظر می‌گیریم. اما این راهکار همواره به درستی عمل نمی‌کند. برای مثال، لیست تمام کلیدواژه‌های نمایگر بخش مراجع مشخص نیست و ممکن است از کلیدواژه‌ای برای عنوان این بخش استفاده شود که در لیست در نظر گرفته شده وجود نداشته باشد. "اطلاعات کتابشناختی"، "مآخذ" و ... مثال‌هایی دیگر از کلیدواژه‌های به کار رفته، برای مشخص کردن بخش مراجع پارساها هستند. علاوه‌براین، رویکرد فوق با این فرض به درستی عمل می‌کند که بخش مراجع، بخش آخر یک پارسا باشد. اما این مساله همواره صحیح نیست و در بسیاری از موارد یک پارسا حاوی ضمائم، جداول یا اشکالی است که بعد از مراجع آورده شده‌اند. با توجه به این موارد، رویکرد ساده مطرح شده در بالا، رویکرد مناسبی جهت استخراج رشته‌های مرجع از یک پارسا نیست.

یک رویکرد دیگر، توجه به ویژگی‌های پاراگراف‌های نمایانگر رشته مرجع، مانند موجود بودن سال در آن‌ها، کوتاه بودن آن‌ها و ... است. بدین منظور، تمام پاراگراف‌های متن پارسا، از ابتدا تا انتها، برای دربرداشتن ویژگی‌های موردنظر مورد بررسی قرار می‌گیرند. با توجه به زیاد بودن ویژگی‌های مورداستفاده و سختی تعیین معیارهای رشته‌مرجع بودن یک پاراگراف، در این پژوهش، از رویکرد یادگیری ماشین استفاده کرده و با استفاده از روش L2-SVM، دسته‌بندی ارائه می‌کنیم که قادر به یادگیری ویژگی‌های رشته‌های مرجع بوده و بتواند رشته‌های مرجع را از سایر پاراگراف‌های متن تشخیص دهد. دسته‌بند ارائه شده که آن را ExtCits نامیده‌ایم، یک دسته‌بند دو کلاسه است که با ورودی یک پاراگراف، یکی از برچسب‌های "رشته مرجع" و "پاراگراف معمولی" را به آن اختصاص می‌دهد.

ویژگی‌های به کار رفته در دسته‌بند:

در این قسمت مجموعه ویژگی‌های در نظر گرفته شده در دسته‌بند پیشنهادی توصیف خواهد شد. برخی از ویژگی‌های استفاده شده در دسته‌بند پیشنهادی از موارد ارائه شده در (Gupta et al., 2009) هستند.

ویژگی‌های در نظر گرفته شده عبارتند از:

۱- شروع یا عدم شروع پاراگراف با کاراکترهای "[" یا "]".
۲- تعداد کلمات موجود در پاراگراف.
۳- نسبت تعداد کاراکترها به تعداد جداکننده‌های موجود در پاراگراف. لیست جداکننده‌های در نظر گرفته شده عبارت است از {"،"، "."، ":"، "؛"، "("، ")"، "»"، "«"، "؟"، "?"، "-"، ","، ";"}.
۴- نسبت تعداد اعداد موجود در پاراگراف به تعداد کلمات پاراگراف.
۵- تعداد عدد نمایانگر سال موجود در پاراگراف. اعداد بین ۱۰۰۰ تا ۲۰۲۰ به عنوان اعداد نمایانگر سال در نظر گرفته شده‌اند.
۶- وجود یا عدم وجود مشخص‌کننده شماره صفحه در لیست واحدهای پاراگراف که مجموعه {"ص"، "صص"، "صفحات"، "صفحه"، "صفحه‌های"، "p"، "pp"، "page"، "pages"} به عنوان مشخص‌کننده‌های شماره صفحه در نظر گرفته شده‌اند. لازم به ذکر است که در اینجا، منظور از واحد، لغت است.
۷- وجود یا عدم وجود مشخص‌کننده شماره‌ی دوره در لیست واحدهای پاراگراف که مجموعه {"دوره"، "دوره‌ی"، "سال"، "س"، "د"، "v"، "vol"، "volume"} به عنوان مشخص‌کننده‌های شماره‌ی دوره در نظر گرفته شده‌اند.
۸- وجود یا عدم وجود مشخص‌کننده شماره در لیست واحدهای پاراگراف که مجموعه {"شماره"، "شماره‌ی"، "شماره‌های"، "ش"، "issue"، "number"، "num"،"n"} به عنوان مشخص‌کننده‌های شماره در دوره در نظر گرفته شده‌اند.
۹- وجود یا عدم وجود مشخص‌کننده شماره چاپ در لیست واحدهای پاراگراف که مجموعه {"چاپ"، "چ"، "چ."، "ed"، "edition"} به عنوان مشخص‌کننده‌های شماره چاپ در نظر گرفته شده‌اند.

۱۰-وجود یا عدم وجود مشخص‌کننده شماره جلد در لیست واحدهای پاراگراف که مجموعه {"جلد"، "ج"، "ج."} به عنوان مشخص‌کننده‌های شماره جلد در نظر گرفته شده‌اند.

۱۱-تعداد واحد مشخص‌کننده نام یک دانشگاه در لیست واحدهای پاراگراف. مجموعه مشخص‌کننده‌های نام یک دانشگاه به صورت زیر در نظر گرفته شده‌اند {"دانشگاه"، "پژوهشگاه"، "دانشکده"، "پژوهشکده"، "پیام"، "نور"، "آزاد"، "گروه"، "استان"، "ایران"، "شهید"، "university"، "faculty"، "center"، "department"}.

۱۲-تعداد واحد مشخص‌کننده ترجمه بودن یک اثر در لیست واحدهای پاراگراف. مجموعه مشخص‌کننده‌ها به صورت {"ترجمه"، "ترجمه‌ی"، "مترجم"، "به‌کوشش"، "بامقدمه"، "بامقدمه‌ی"، "تصحیح"} در نظر گرفته شده‌اند.

۱۳-تعداد واحد مشخص‌کننده پارسا بودن یک اثر در لیست واحدهای پاراگراف. مجموعه مشخص‌کننده‌ها به صورت {"استاد"، "راهنما"، "پایان‌نامه"، "رساله"، "کارشناسی"، "کارشناسی‌ارشد"، "دکتری"، "دکترا"، "رشته"، "رساله‌ی"، "راهنمایی"، "thesis"، "dissertation"، "phd"، "master"} در نظر گرفته شده‌اند.

۱۴-تعداد واحد مشخص‌کننده نام انتشارات در لیست واحدهای پاراگراف. مجموعه مشخص‌کننده‌ها به صورت {"انتشارات"، "نشر"، "ناشر"، "الانتشار"، "چاپ"، "جهاد"، "publication"، "press"، "inc"} در نظر گرفته شده است.

۱۵-تعداد نام شهر معروف ذکر شده در لیست واحدهای پاراگراف. لیست شهرهای معروف به صورت زیر در نظر گرفته شده عبارت است از {"تهران"، "اصفهان"، "شیراز"، "تبریز"، "یزد"، "بیروت"، "نجف"، "قم"، "newyork"، "London"، "united"، "states"، "us"، "uk"، "Britain"، "Melbourne"}.

۱۶- وجود مشخص‌کننده نام نشریه به عنوان یک واحد در پاراگراف. مجموعه مشخص‌کننده‌ها به صورت {"فصلنامه"، "پژوهشنامه"، "مجله"، "دوفصلنامه"، "نشریه"، "دوماهنامه"، "ماهنامه"، "پژوهش‌نامه"، "فصل‌نامه"، "دوفصل‌نامه"، "ماه‌نامه"، "پژوهش"، "تحقیقات"، "تازه‌های"، "مطالعات"، "journal"، "j"، "research"، "international"، "int"، "letters"، "let"، "science"، "sciences"، "sci"، "review"، "advances"، "bulletin"} در نظر گرفته شده است.

۱۷- وجود مشخص‌کننده نام همایش به عنوان یک واحد در پاراگراف. مجموعه مشخص‌کننده‌ها به صورت زیر در نظر گرفته شده است: {"همایش"، "کنفرانس"، "ایران"، "ملی"، "بین‌المللی"، "کنگره"، "سمینار"، "انجمن"، "اولین"، "نخستین"، "یکمین"، "دومین"، "سومین"، "چهارمین"، "پنجمین"، "ششمین"، "هفتمین"، "هشتمین"، "نهمین"، "دهمین"، "یازدهمین"، "دوازدهمین"، "سیزدهمین"، "چهاردهمین"، "پانزدهمین"، "شانزدهمین"، "هفدهمین"، "هجدهمین"، "نوزدهمین"، "خلاصه"، "مقالات"، "international"، "conference"، "symposium"، "congress"، "proceedings"، "annual"}.

٤-۲-۱. دسته‌بند ExtCits:

- **آموزش:**

a. متون ورودی به مجموعه‌ای از پاراگراف‌ها و هر پاراگراف به دنباله‌ای از واحدها تجزیه شده و بردار ویژگی متناظر با هر پاراگراف محاسبه می‌شود.

b. دسته‌بند SVM با استفاده از بردارهای ویژگی استخراج شده و برچسب‌های صحیح اختصاص‌یافته به هر پاراگراف آموزش داده می‌شود.

- **تعیین رشته مرجع بودن یا نبودن یک پاراگراف جدید:**

a. پاراگراف جدید به دنباله‌ای از واحدها تقسیم شده و بردار ویژگی متناظر با آن محاسبه می‌شود.

b. با استفاده از دسته‌بند آموزش دیده، برچسب این پاراگراف (رشته مرجع یا پاراگراف معمولی) به دست می‌آید.

٤-۲-۲. ارزیابی دسته‌بند ExtCits

برای ارزیابی روش ارائه شده از مجموعه داده طراحی شده بدین‌منظور و "روش اعتبارسنجی متقابل ۱۰-سطحی"[۱] استفاده شده است. مجموعه داده ایجاد شده به منظور بررسی کیفیت دسته‌بند پیشنهادی شامل پاراگراف‌های تمام متن ۱۰ پارسا شامل ۱۳۷۳۹ پاراگراف می‌باشد که به هر پاراگراف از این مجموعه داده یک برچسب از مجموعه برچسب‌های {رشته مرجع، پاراگراف معمولی} اختصاص یافته است. نتایج پیاده‌سازی دسته‌بند پیشنهادی روی

[۱] 10-fold cross validation

مجموعه داده موردنظر نشان می‌دهد که میانگین دقت، فراخوانی و F1 در روش ارائه شده برابر با ۱۰۰٪ بوده و دسته‌بند حاصل، تقریبا به ازای تمام داده‌های آزمایش، برچسب درست را تشخیص داده است. این نتایج به صورت جزئی و برای هر برچسب به صورت مجزا در جدول شماره ۱ گزارش شده‌اند.

جدول ۱. نتایج ارزیابی دسته‌بند پیشنهادی برای شناسایی رشته‌های مرجع در تمام متن پارساها.

برچسب	دقت	فراخوانی	F1	تعداد
رشته مرجع	۱,۰۰	۱,۰۰	۱,۰۰	۱۱۳۱۶
پاراگراف معمولی	۱,۰۰	۰,۹۹	۱,۰۰	۲۴۲۳
مجموع	۱,۰۰	۱,۰۰	۱,۰۰	۱۳۷۳۹

٤-۲-۳. نمونه‌هایی از نتایج به دست آمده با استفاده از دسته‌بند ExtCits

در این قسمت، نمونه‌هایی از نتایج حاصل از اعمال دسته‌بند ExtCits روی تعدادی پاراگراف ورودی ارائه خواهد شد. نمونه‌های موردنظر در جدول ۲ مشخص شده‌اند که در آن، پس‌زمینه نمونه‌های اشتباه تشخیص داده شده رنگی است. لازم به ذکر است که با توجه به رویکرد در نظر گرفته شده، اغلب نمونه‌های اشتباه به دلیل نگارش رشته‌های مرجع در دو پاراگراف به وجود آمده‌اند.

جدول ۲. نمونه‌هایی از نتایج به دست آمده با استفاده از دسته‌بند ExtCits

پاراگراف	برچسب اختصاص یافته
براهیمیقوام، صغری (۱۳۷۱). تعیین ضرایب پایانی و همبستگی سه مفهوم حمایت اجتماعی، عزت نفس و مرجع کنترل بر روی دانشجویان و دانشآموزان تهران، پایاننامه کارشناسی ارشد، دانشگاه آزاد اسلامی واحد تهران.	رشته مرجع
بشارت، محمدعلی (۱۳۸۱). ابعاد کمالگرایی در بیماران افسرده و مضطرب، مجله علوم شناختی، ص۲۶۳-۲۴۸	رشته مرجع
حافظی، فریبا، احدی، حسن، عنایتی، میرصلاحالدین و نجاریان، بهمن (۱۳۸۶). رابطهی علی استرس چالش، استرس مانع پیشرفت، فرسودگی و انگیزش یادگیری با عملکرد تحصیلی در دانشجویان دانشگاه آزاد اسلامی اهواز. دانش و پژوهش در روانشناسی کاربردی، شمارهی سی و دوم، تابستان ۱۳۸۶.	رشته مرجع
سرمد، زهره؛ بازرگان، عباس؛ و حجازی، الهه (۱۳۹۱). روشهای تحقیق در علوم رفتاری. کتاب گزیده	رشته مرجع
لیوارجانی، شعله (۱۳۷۵). بررسی عملی بودن، اعتبار، روایی و هنجاریابی مقیاس حمایت اجتماعی فلمینگ، باوم، گیریل و گچل در میان دانشآموزان مقطع دبیرستان. پایاننامه	رشته مرجع

کارشناسی ارشد دانشگاه علامه طباطبایی.	
مهرابیزاده هنرمند، مهناز و وردی، مینا (۱۳۸۲). کمالگرایی مثبت، کمالگرایی منفی. اهواز: رسش.	رشته مرجع
Balogun, J. A, Helgemoe, S, Pellegrini, E & Hoeberlein, T, (1996). Academic performance is not a viable determinant of physical therapy students' burnout, Perceptual and Motor Skills, 83, 21-22.	رشته مرجع
Bandura, A. (1997). Self-efficacy: The exercise og control. New York: W. H. Freeman.	رشته مرجع
Comrey, A. L., & Lee, H. B. (1992). A first course in factor analysis (2nded.). Hillsdale, NJ: Erlbaum.	رشته مرجع
Goplerud, Eric N. (1980). Social support and stress during the first year of graduate school.	رشته مرجع
Professional Psychology, Vol 11(2), Apr 1980, 283-290. http://dx.doi.org/10.1037/0735-7028.11.2.283	رشته مرجع
Maslach, C. (1982). Burnout: The cost of caring. Englewood Cliffs, NJ: Prentice Hall.	رشته مرجع
Perrewe, P. L., Hochwarter, W. A., Rossi, A. M., Wallace, A., Maignan, I., Castro, S. L., Ralston, D. A., Westman, M., Vollmer, G., Tang, M., Wan, P., & Van Deusen, C. A. (2002). Are work stress relationships universal? A nine-region examination of role stressors, general self-efficacy, and burnout. Journal of International Management, 8(2), 163-187.	رشته مرجع

٤-۳- تعیین زبان رشته مرجع

تجزیه و تطبیق رشته‌های مرجع در زبان‌های مختلف متفاوت بوده و نیازمند الگوریتم‌های منحصر به خود می‌باشند. با توجه به این مساله، برای تجزیه و تطبیق مناسب رشته‌های مرجع، باید در ابتدا زبان رشته‌های مرجع ورودی تشخیص داده شود. همانطور که بیان شد، در این پژوهش فرض می‌شود که یک رشته مرجع ورودی به یکی از دو زبان فارسی یا انگلیسی باشد. با توجه به این فرض، برای تعیین زبان یک رشته مرجع، به طور ساده از شمارش تعداد کاراکترهای فارسی و انگلیسی آن رشته مرجع استفاده می‌شود. جزئیات این الگوریتم ساده که آن را LangDet نامیده‌ایم، در ادامه بیان می‌شود.

الگوریتم LangDet

۱- تعداد حروف فارسی موجود در کاراکترهای رشته مرجع شمارش می‌شود. در صورتی که این تعداد بیش از نیمی از تعداد کل کاراکترهای موجود در رشته باشد، زبان رشته مرجع، فارسی تشخیص داده می‌شود.

۲- تعداد حروف انگلیسی موجود در کاراکترهای رشته مرجع شمارش می‌شود. در صورتی که این تعداد بیش از نیمی از تعداد کل کاراکترهای موجود در رشته باشند، زبان رشته مرجع، انگلیسی تشخیص داده می‌شود.

٤-٤- تعیین نوع رشته‌های مرجع

یکی از موارد موردنیاز در این پژوهش، تعیین نوع اسناد ذخیره شده در جدول اسناد است. منابع ورودی به این جدول از دو نوع هستند: ۱- پارسا هستند که نوع آن‌ها مشخص است، ۲- از رشته‌های مرجع موجود در پارساها به دست می‌آیند که همانطور که مطرح شد، فرض می‌کنیم از یکی از ۴ نوع مقاله نشریه، مقاله همایش، پارسا یا کتاب باشند. در این پژوهش، از رویکرد مبتنی بر یادگیری ماشین برای دسته‌بندی رشته‌های مرجع در یکی از این ۴ دسته استفاده می‌کنیم. با توجه به استفاده از ویژگی‌های منحصر به زبان در دسته‌بند مورد استفاده، در ادامه و برای تعیین نوع رشته‌های مرجع در دو زبان فارسی و انگلیسی، دو دسته‌بند مجزا ارائه می‌کنیم که به ترتیب آن‌ها را PerTypeDet و EngTypeDet نامیده‌ایم.

٤-٤-۱. تعیین نوع رشته‌های مرجع به زبان فارسی

در این بخش، با استفاده از روش OSVM دسته‌بندی ارائه می‌کنیم که قادر به تشخیص نوع رشته‌های مرجع به زبان فارسی باشد. همانطور که قبلا بیان کردیم، در اینجا فرض می‌شود که نوع رشته مرجع یکی از چهار مورد مقاله نشریه، مقاله همایش، پارسا و کتاب باشد.

ویژگی‌های به کار رفته در دسته‌بند:

در این قسمت مجموعه ویژگی‌های در نظر گرفته شده در دسته‌بند پیشنهادی توصیف خواهد شد. ویژگی‌های در نظر گرفته شده با توجه به تجربه پژوهشگر و ویژگی‌های استفاده شده در (پاک‌نیت و همکاران، ۱۳۹۷) تعیین شده‌اند:

۱- وجود یا عدم وجود کاراکتر ":" در رشته مرجع.

۲- تعداد اعداد موجود در رشته مرجع.

۳- وجود یا عدم وجود مشخص‌کننده شماره صفحه در لیست واحدهای رشته مرجع. مجموعه مشخص‌کننده‌های صفحه به صورت زیر در نظر گرفته شده است:{"ص"، "صص"، "صفحات"، "صفحه"، "صفحه‌های}. لازم به ذکر است که در اینجا، منظور از واحد لغت است.

۴- وجود یا عدم وجود واحدی با قالب شماره صفحه در لیست واحدهای رشته مرجع. بدین منظور در صورتی که واحد با مشخص‌کننده‌های شماره صفحه آغاز شده باشد، مشخص‌کننده از ابتدای آن حذف شده و در ادامه در صورتی که واحد شامل "-" و تعدادی عدد باشد، به عنوان واحدی در قالب شماره صفحه در نظر گرفته می‌شود.

۵- وجود یا عدم وجود مشخص‌کننده دوره و شماره در لیست واحدهای رشته مرجع که از مجموعه {"دوره"، "دوره‌ی"، "سال"، "س"، "د"} به عنوان مجموعه مشخص‌کننده‌های دوره و از {"شماره"، "شماره‌ی"، "ش"} به عنوان مجموعه‌مشخص‌کننده‌های شماره استفاده شده است.

۶- وجود یا عدم وجود مشخص‌کننده شماره جلد و شماره چاپ در لیست واحدهای رشته مرجع که از {"جلد"، "ج"، "ج."} به عنوان مجموعه مشخص‌کننده‌های شماره جلد و از {"چاپ"، "چ"، "چ."} به عنوان مجموعه مشخص‌کننده‌های شماره چاپ استفاده شده است.

۷- وجود یا عدم وجود نام یکی از ماه‌ها یا فصل‌های سال در لیست واحدهای رشته مرجع. به طور واضح، مجموعه در نظر گرفته شده بدین منظور عبارت است از{"فروردین"، "اردیبهشت"، "خرداد"، "تیر"، "مرداد"، "شهریور"، "مهر"، "آبان"، "ابان"، "آذر"، "دی"، "بهمن"، "اسفند"، "بهار"، "تابستان"، "پاییز"، "زمستان"}.

۸- وجود یا عدم وجود یکی از مشخص‌کننده‌های نام دانشگاه در لیست واحدهای رشته مرجع. مجموعه {"دانشگاه"، "پژوهشگاه"، "دانشکده"، "پژوهشکده"، "پیام"، "نور"، "آزاد"، "گروه"، "استان"، "ایران"، "شهید"} به عنوان مجموعه مشخص‌کننده‌های نام دانشگاه مورد استفاده قرار گرفته است.

۹- وجود یا عدم وجود یکی از مشخص‌کننده‌های ترجمه بودن سند در لیست واحدهای رشته مرجع که مجموعه مشخص‌کننده‌های ترجمه بودن یک رشته مرجع به صورت زیر در نظر گرفته شده است: {"ترجمه"، "ترجمه‌ی"، "مترجم"، "به‌کوشش"، "بامقدمه"، "بامقدمه‌ی"، "تصحیح"}.

۱۰-تعداد واحد موجود در رشته مرجع که مشخص‌کننده پارسا هستند. مجموعه مشخص‌کننده‌های پارسا به صورت زیر در نظر گرفته شده است: {"استاد"، "راهنما"، "راهنمایی"، "پایان‌نامه"، "رساله"، "رساله‌ی"، "کارشناسی"، "کارشناسی‌ارشد"، "دکتری"، "دکترا"}.

۱۱-وجود یا عدم وجود مشخص‌کننده انتشارات در لیست واحدهای رشته مرجع که بدین منظور از مجموعه زیر به عنوان مشخص‌کننده‌های انتشارات استفاده شده است: {"انتشارات"، "نشر"، "ناشر"، "الانتشار"، "چاپ"، "جهاد"}.

۱۲-وجود نام یکی از شهرهای مهم در لیست واحدهای رشته مرجع که لیست شهرهای مهم مورد استفاده به صورت زیر است: {"تهران"، "اصفهان"، "شیراز"، "تبریز"، "یزد"، "بیروت"، "نجف"، "قم"}.

۱۳-وجود یا عدم وجود یکی از مشخص‌کننده‌های نام نشریه در لیست واحدهای رشته مرجع که از {"فصلنامه"، "پژوهشنامه"، "مجله"، "دوفصلنامه"، "نشریه"، "نامه"، "دوماهنامه"، "ماهنامه"، "پژوهش"، "تحقیقات"، "تازه‌های"، "مطالعات"} به عنوان مجموعه مشخص‌کننده‌های نام نشریه استفاده شده است.

۱۴-وجود یا عدم وجود یکی از مشخص‌کننده‌های همایش در لیست واحدهای رشته مرجع که از مجموعه {"همایش"، "کنفرانس"، "ایران"، "ملی"، "بین‌المللی"، "کنگره"، "کنگره‌ی"، "سمینار"، "انجمن"، "اولین"، "نخستین"، "یکمین"، "دومین"، "سومین"، "چهارمین"، "پنجمین"، "ششمین"، "هفتمین"، "هشتمین"، "نهمین"، "دهمین"، "یازدهمین"، "دوازدهمین"، "سیزدهمین"، "چهاردهمین"، "پانزدهمین"، "شانزدهمین"، "هفدهمین"، "هجدهمین"، "نوزدهمین"، "خلاصه"، "مقالات"} بدین منظور استفاده شده است..

دسته‌بند PerTypeDet

- **آموزش:**

۱- هر رشته مرجع به دنباله‌ای از واحدها تقسیم شده و بردار ویژگی متناظر با هر رشته مرجع محاسبه می‌شود.

۲- دسته‌بند OSVM با استفاده از بردارهای ویژگی استخراج شده و برچسب‌های صحیح اختصاص‌یافته به هر رشته مرجع آموزش داده می‌شود.

- **تعیین نوع یک رشته مرجع جدید:**

۱- رشته مرجع ورودی به دنباله‌ای از واحدها تقسیم شده و بردار ویژگی متناظر با رشته مرجع جدید محاسبه می‌شود.

۲- با استفاده از دسته‌بند آموزش دیده، نوع این رشته تعیین می‌شود.

ارزیابی دسته‌بند PerTypeDet

برای ارزیابی روش ارائه شده از مجموعه داده طراحی شده در (پاک‌نیت و همکاران ۱۳۹۷) و روش اعتبارسنجی متقابل ۱۰-سطحی استفاده شده است. مجموعه داده ایجاد شده به منظور بررسی کیفیت دسته‌بند پیشنهادی شامل ۹۰۰ رشته مرجع (۳۵۹ کتاب، ۳۹۷ مقاله چاپ شده در نشریه، ۶۰ مقاله ارائه شده در همایش و ۸۴ پارسا) می‌باشد. نتایج پیاده‌سازی دسته‌بند پیشنهادی روی مجموعه داده موردنظر نشان می‌دهد که میانگین دقت، فراخوانی و F1 در روش ارائه شده برابر با ۹۵٪ است. این نتایج به صورت جزئی و برای هر نوع از رشته‌های مرجع به صورت مجزا در جدول شماره ۳ گزارش شده است.

جدول ۳. نتایج ارزیابی دسته‌بند پیشنهادی برای تعیین نوع رشته‌های مرجع به زبان فارسی

برچسب	دقت	فراخوانی	F1	تعداد
کتاب	۰,۹۳	۰,۹۶	۰,۹۵	۲۵۰
همایش	۰,۹۸	۰,۹۲	۰,۹۵	۲۵۰
نشریه	۰,۹۵	۰,۹۷	۰,۹۶	۲۵۰
پارسا	۰,۹۹	۰,۹۹	۰,۹۹	۲۵۰
مجموع	۰,۹۶	۰,۹۶	۰,۹۶	۱۰۰۰

نمونه‌هایی از نتایج به دست آمده با استفاده از دسته‌بند PerTypeDet

در این قسمت و در جدول ۴، نمونه‌هایی از نتایج حاصل از اعمال دسته‌بند PerTypeDet روی تعدادی رشته مرجع ورودی ارائه خواهد شد.

جدول ٤. نمونه‌هایی از نتایج به دست آمده با استفاده از دسته‌بند PerTypeDet

رشته مرجع	برچسب اختصاص یافته
مقدسی، ثریا (۱۳۹۱). رابطه جهتگیری هدف و حمایت اجتماعی ادراکشده با فرسودگی تحصیلی دانشجویان دانشگاه شهید مدنی آذربایجان. پایاننامه کارشناسی ارشد، دانشگاه	پارسا

شهید مدنی آذربایجان	
_ قشیری، عبدالکریم بن هوازن، (۱۳۸۱)، رساله قشیریه، ترجمه ابوعلی حسن بن احمد عثمانی، تصحیح بدیع ازمان فروزانفر، تهران: انتشارات علمی و فرهنگی.	کتاب
سواری، ر. ۱۳۹۱. مطالعه مقدماتی برخی ویژگی‌های تولیدمثلی دو گونه بارناکل بین جزر و مدی Microeuraphia permitini و Amphibalanus Amphitrite در سواحل بندرعباس خلیج‌فارس. پایان‌نامه کارشناسی ارشد، زیست‌شناسی دریا- جانوران دریا، دانشگاه هرمزگان، ۱۲۰ صفحه.	پارسا
خامنهای، سیدعلی، «بررسی ابعاد حکومت اسلامی، حکومت در اسلام»، مقالات سومین و چهارمین کنفرانس اندیشه اسلامی، تهران امیر کبیر، ۱۳۶۷ش.	مقاله همایش
نوذری، شعله.۱۳۸۳. رهنمودهای طراحی فضای باز مسکونی. مجله صفه شماره ۳۹	مقاله نشریه
نجارزادگان، فتح الله، کارکردهای عقل در تفسیر وحی از نگاه فریقین، بهار و تابستان ۱۳۸۴، اندیشه های فلسفی، سال اول، ش دوم، ۱۳۵-۱۵۰	مقاله نشریه
گلدوزیان ، ایرج ، بایسته های حقوق جزای عمومی ، جلد ۱-۲-۳ ، نشر میزان ، چاپ چهارم ، ۱۳۸۰.	کتاب
عبدالهی، بیژن؛ کریمیان، حیدر؛ نامداریپژمان، مهدی. (۱۳۹۳). ارتباط تعهد سازمانی و معنویت در محیط کار با رفتار اخلاقی. فصلنامه اخلاق در علوم و فنّاوری، ۹(۴): ۶۰-۵۱.	مقاله نشریه
رفالیان، غزل (۱۳۸۹)، کاربرد روش های طراحی مقداری در فرآیند طراحی معماری با تأکید به هماهنگی معماری و سازه، پایان نامه کارشناسی ارشد رشته معماری، دانشگاه تربیت مدرس، دانشکده هنر و معماری.	پارسا
- دین‌محمدی، مصطفی، دل‌انگیزان، سهراب، صادقی، زین‌العابدین(۱۳۸۴)، خوشه‌بندی فضایی صنایع با فناوری برتر و تاثیر آن بر توسعه فناوری، دومین همایش دوسالانه آموزش عالی و اشتغال، تهران.	مقاله همایش

٤-٤-٢. تعیین نوع رشته‌های مرجع به زبان انگلیسی

در این بخش، با استفاده از روش OSVM دسته‌بندی ارائه می‌کنیم که قادر به تشخیص نوع رشته‌های مرجع به زبان انگلیسی باشد.

ویژگی‌های به کار رفته در دسته‌بند:

در این قسمت مجموعه ویژگی‌های در نظر گرفته شده در دسته‌بند پیشنهادی توصیف خواهد شد. ویژگی‌های در نظر گرفته شده با توجه به تجربه پژوهشگر تعیین شده‌اند:

۱- وجود یا عدم وجود کاراکتر ":" در رشته مرجع.

۲- تعداد اعداد موجود در رشته مرجع.

۳- وجود یا عدم وجود مشخص‌کننده شماره صفحه در لیست واحدهای رشته مرجع. مجموعه مشخص‌کننده‌های شماره صفحه به صورت {"p"، "pp"، "page"، "pages"} در نظر گرفته شده است. لازم به ذکر است که در اینجا، منظور از واحد، لغت است.

۴- وجود یا عدم وجود واحدی با قالب شماره صفحه در لیست واحدهای رشته مرجع. واحدهای با قالب شماره صفحه همانند مورد مشابه در بخش پیش مشخص می‌شوند.

۵- وجود یا عدم وجود مشخص‌کننده دوره یا شماره در لیست واحدهای رشته مرجع که مجموعه {"v"، "vol"، "volume"} به عنوان مجموعه مشخص‌کننده‌های دوره و {"issue"، "number"، "num"،"n"} به عنوان مجموعه مشخص‌کننده شماره در نظر گرفته شده است.

۶- وجود یا عدم وجود مشخص‌کننده شماره چاپ در لیست واحدهای رشته مرجع. مجموعه مشخص‌کننده‌های در نظر گرفته شده عبارت است از {"ed"، "edition"}.

۷- وجود یا عدم وجود نام یکی از ماه‌های میلادی یا فصل‌های سال به زبان انگلیسی در لیست واحدهای رشته مرجع. {"january"، "february"، "march"، "april"، "may"، "june"، "july"، "august"، "september"، "october"، "november"، "December"، "jan"، "feb"، "mar"، "apr"، "may"، "jun"، "jul"، "aug"، "sep"، "sept"، "oct"، "nov"، "dec"، "spring"، "summer"، "fall"، "winter"} به عنوان مجموعه ماه‌ها و فصل‌ها در نظر گرفته شده است.

۸- وجود یا عدم وجود یکی از مشخص‌کننده‌های نام دانشگاه در لیست واحدهای رشته مرجع که از {"university"، "faculty"، "center"، "department"} به عنوان مجموعه مشخص‌کننده‌های نام دانشگاه استفاده شده است.

۹- وجود یا عدم وجود یکی از مشخص‌کننده‌های {"in"، "eds"} در لیست واحدهای رشته مرجع.

۱۰- تعداد واحد از لیست واحدهای رشته مرجع که در مجموعه مشخص‌کننده‌های پارسا وجود دارند. مجموعه مشخص‌کننده‌های پارسا به صورت زیر در نظر گرفته شده است: {"thesis"، "dissertation"، "phd"، "master"}.

۱۱-تعداد واحدهای رشته مرجع که در نام انتشارات‌های مهم ذکر شده‌اند. برای نام انتشارات‌های مهم از مجموعه انتشارات با رتبه A و رتبه B گزارش شده در آدرس https://research.usp.ac.fj/wp-content/uploads/2013/09/2016-Ranking-of-Academic-Publishers-v1.0.pdf استفاده شده است.

۱۲-وجود نام یکی از شهرهای مهم در لیست واحدهای رشته مرجع که لیست شهرهای مهم مورد استفاده به صورت زیر است: {"newyork"، "london"، "united"، "states"، "us"، "uk"، "britain"، "Melbourne"}.

۱۳-وجود یکی از مشخص‌کننده‌های نشریه در لیست واحدهای رشته مرجع. مجموعه مشخص‌کننده‌های در نظر گرفته شده بدین منظور عبارت است از {"journal"، "j"، "research"، "international"، "int"، "letters"، "lett"، "science"، "sciences"، "sci"، "review"، "procedia"، "advances"، "bulletin"}.

۱۴-وجود یکی از مشخص‌کننده‌های همایش در لیست واحدهای رشته مرجع. مجموعه مشخص‌کننده‌های در نظر گرفته شده بدین منظور عبارت است از {"international"، "conference"، "symposium"، "congress"، "proceedings"، "annual"}.

دسته‌بند EngTypeDet

- **آموزش:**

۱- رشته‌های مرجع ورودی به دنباله‌ای از واحدها تقسیم شده و بردار ویژگی متناظر با هر رشته مرجع محاسبه می‌شود.

۲- دسته‌بند OSVM با استفاده از بردارهای ویژگی استخراج شده و برچسب‌های صحیح اختصاص‌یافته به هر رشته مرجع آموزش داده می‌شود.

- **تعیین نوع یک رشته مرجع جدید:**

۱- رشته مرجع جدید به دنباله‌ای از واحدها تقسیم شده و بردار ویژگی متناظر با رشته مرجع جدید محاسبه می‌شود.

۲- با استفاده از دسته‌بند آموزش دیده، نوع این رشته تعیین می‌شود.

ارزیابی دسته‌بند EngTypeDet

برای ارزیابی روش ارائه شده از مجموعه داده طراحی شده بدین منظور و روش اعتبارسنجی متقابل ۱۰-سطحی استفاده شده است. مجموعه داده ایجاد شده به منظور بررسی کیفیت دسته‌بند پیشنهادی شامل ۲۹۲ رشته مرجع (۵۱ کتاب، ۵۹ مقاله چاپ شده در نشریه، ۱۷۹ مقاله ارائه شده در همایش و ۳ پارسا) می‌باشد. نتایج پیاده‌سازی دسته‌بند پیشنهادی روی مجموعه داده موردنظر نشان می‌دهد که میانگین دقت، فراخوانی و F1 در روش ارائه شده برابر با ۸۵٪ است. این نتایج به صورت جزئی و برای هر نوع از رشته‌های مرجع به صورت مجزا در جدول شماره ۵ گزارش شده‌اند.

جدول ۵. نتایج ارزیابی دسته‌بند پیشنهادی برای تعیین نوع رشته‌های مرجع به زبان انگلیسی

برچسب	دقت	فراخوانی	F1	تعداد
کتاب	۰٫۸۵	۰٫۹۰	۰٫۸۸	۲۵۰
همایش	۰٫۹۷	۰٫۸۷	۰٫۹۲	۲۵۰
نشریه	۰٫۸۸	۰٫۹۴	۰٫۹۱	۲۵۰
پارسا	۰٫۹۷	۰٫۹۶	۰٫۹۷	۲۵۰
مجموع	۰٫۹۲	۰٫۹۲	۰٫۹۲	۱۰۰۰

نمونه‌هایی از نتایج به دست آمده با استفاده از دسته‌بند EngTypeDet

در این قسمت، نمونه‌هایی از نتایج حاصل از اعمال دسته‌بند EngTypeDet روی تعدادی رشته مرجع ورودی ارائه خواهد شد. نمونه‌های موردنظر در جدول ۶ ارائه شده‌اند که در آن، پس‌زمینه نمونه‌های اشتباه تشخیص داده شده رنگی است.

جدول ٦. نمونه‌هایی از نتایج به دست آمده با استفاده از دسته‌بند EngTypeDet

رشته مرجع	برچسب اختصاص یافته
Bandura, A. (1982). Self-efficacy mechanism in human agency. American Psychologist, 37, 122-147.	مقاله نشریه

مقاله نشریه	Bakker, A.B., Demerouti, E. & Schaufeli, W.B. (2003). Dual processes at work in a call centre: An application of the Job Demands – Resources model. European Journal of Work and Organizational Psychology, 12, 393-417.
کتاب	Maslach, C. (1982). Burnout: The cost of caring. Englewood Cliffs, NJ: Prentice Hall.
پارسا	Clark, K. D. (2010). The Relationship of Perceived Stress and Self-Efficacy Among Correctional Employees in Close-Security and Medium-Security-Level Institutions. Degree of Doctor of Philosophy,Psychology, Walden University.
پارسا	House, J.S., 1981. Work Stress and Social Support. Addison-Wesley, Reading, MA.
مقاله نشریه	Birner, B. J. (1997). The linguistic realization of inferrable information. Language & Communication, 17(2), 133-147.
کتاب	Chafe, W. (1976). Givenness, Contrastiveness, Definiteness, Subjects, Topics and Point of View. In Charles Li, (ed.), Subjects and Topics, 27-55, New York: Academic press.
کتاب	Gosling, D., & Maitland, B., 1976: Design and planning of retail systems. New York: Whitney Library of Design.
مقاله همایش	Hwang, Y., & Yi, M. (2002). Predicting the use of web- based information - system: Intrinsic motivation and self-efficacy, Eighth Americas Conference on Informational systems.

٤-٥- تجزیه رشته‌های مرجع

همانطور که بیان شد، یک رشته مرجع را می‌توان به عنوان مجموعه‌ای از مولفه‌ها مانند نام نویسندگان، عنوان، محل نشر، سال نشر، شماره صفحات و ... در نظر گرفت. در حالیکه تجزیه رشته‌های مرجع موجود در انتهای یک مدرک علمی توسط کاربر انسانی به راحتی انجام‌پذیر است، تنوع موجود در قالب‌های نگارش رشته‌های مرجع در کنار اشتباهات رخ داده توسط نویسندگان در نگارش این رشته‌ها، از دشواری‌های موجود در خودکارسازی این عملیات هستند. با توجه به وابستگی روش‌های تجزیه رشته مرجع به زبان، در این بخش دو الگوریتم برای تجزیه رشته‌های مرجع در زبان فارسی و انگلیسی ارائه می‌دهیم. هر دو روش ارائه شده مبتنی بر دسته‌بند OSVM هستند. روش ارائه شده برای تجزیه رشته‌های مرجع در زبان فارسی که آن را PerCitPars نامیده‌ایم، بهبودی از روش ارائه شده در (پاک‌نیت و همکاران ۱۳۹۷) است که در آن، تجزیه مولفه نام نویسندگان به اسامی تک تک نویسندگان نیز صورت می‌گیرد. روش ارائه شده برای تجزیه رشته‌های مرجع به زبان انگلیسی نیز با توجه به روش ارائه شده برای زبان فارسی و همچنین روش ارائه شده در (Zhang et al. 2011) به دست آمده و آن را EngCitPars نامیده‌ایم.

٤-٥-١. تجزیه رشته‌های مرجع در زبان فارسی

در این بخش، برای تجزیه یک رشته مرجع به مولفه‌های سازنده نسخه‌ای بهبودیافته از دسته‌بند پیشنهادی در (پاک‌نیت و همکاران ۱۳۹۷) ارائه می‌کنیم. تفاوت نسخه جدید با نسخه قبل در این است که در نسخه جدید، مولفه نویسندگان یک رشته مرجع با انجام برخی پردازش‌های پسین به لیست نام نویسندگان تجزیه می‌شود. لازم به ذکر است که در این دسته‌بند، یک رشته مرجع به مجموعه‌ای از لغات به عنوان واحدها تقسیم می‌شود و دسته‌بندی در سطح لغت انجام می‌شود. علاوه‌براین، مجموعه‌های مورد استفاده در این بخش برابر با مجموعه‌های مورد استفاده در بخش ۴-۴-۱ هستند.

ویژگی‌های به کار رفته در دسته‌بند:

۱- موقعیت واحد در رشته مرجع.

۲- وجود یا عدم وجود یکی از علائم نقطه‌گذاری {"،"، "."، "؛"، "("، ")"، "\"، "»"، "«"، "؟"، ":"} قبل از واحد.

۳- وجود یا عدم وجود یکی از علائم نقطه‌گذاری فوق پس از واحد.

۴- وجود یا عدم وجود واحد در مجموعه مشخص‌کننده‌های شماره صفحه.

۵- وجود یا عدم وجود واحد در قالب شماره صفحه.

۶- عدد یا غیر عدد بودن واحد.

۷- تعداد واحدهای عددی مشاهده شده قبل از واحد فعلی.

۸- بودن یا نبودن واحد در قالب سال.

۹- بودن یا نبودن واحد در قالب نام یک نویسنده (مخفف نام نویسنده). بدین منظور حضور واحد در مجموعه {"ا."، "آ."، "ب."، "پ."، "ت."، "ث."، "ج."، "چ."، "ح."، "خ."، "د."، "ذ."، "ر."، "ز."، "ژ."، "س."، "ش."، "ص."، "ض."، "ط."، "ظ."، "ع."، "غ."، "ف."، "ق."، "ک."، "گ."، "ل."، "م."، "ن."، "و."، "ه."، "ی."} مورد بررسی قرار می‌گیرد.

۱۰- ختم یا عدم ختم واحد به یکی از پسوندهای نام خانوادگی رایج در زبان فارسی. مجموعه پسوندهای در نظر گرفته شده عبارتند از {"پور"، "فرد"، "نیا"، "زاده"، "نژاد"، "راد"، "فر"، "ان"، "کیا"، "پناه"، "منش"}.

۱۱- وجود یا عدم وجود کاراکتری غیر از کاراکترهای متناظر با حروف فارسی در واحد.

۱۲-وجود یا عدم وجود همزمان حروف الفبای فارسی و عدد در واحد.

۱۳-وجود یا عدم وجود کاراکتری غیر از حروف الفبای فارسی و عدد در واحد.

۱۴-وجود یا عدم وجود واحد در مشخص‌کننده‌های دوره و شماره.

۱۵-وجود یا عدم وجود واحد در مشخص‌کننده‌های شماره چاپ.

۱۶-وجود یا عدم وجود واحد در مجموعه مشخص‌کننده‌های شماره جلد.

۱۷-وجود یا عدم وجود واحد در مجموعه نام ماه‌ها و فصل‌های یک سال.

۱۸-وجود یا عدم وجود واحد در مجموعه اعداد نوشته شده. از نگارش فارسی اعداد کمتر از صد به عنوان مجموعه اعداد نوشته شده استفاده شده است.

۱۹-وجود یا عدم وجود حداقل یک حرف از زبان انگلیسی (چه بزرگ، چه کوچک) در واحد.

۲۰-وجود یا عدم وجود واحد در مجموعه {"دیگران"، "همکاران"}.

۲۱-وجود یا عدم وجود واحد در مجموعه مشخص‌کننده‌های نام دانشگاه.

۲۲-وجود یا عدم وجود واحد در مجموعه مشخص‌کننده‌های ترجمه بودن یک سند.

۲۳-وجود یا عدم وجود واحد در مجموعه مشخص‌کننده‌های پارسا.

۲۴-وجود یا عدم وجود واحد در مجموعه مشخص‌کننده‌های نام انتشارات.

۲۵-وجود یا عدم وجود واحد در لیست شهرهای مهم.

۲۶-وجود یا عدم وجود واحد در مجموعه مشخص‌کننده‌های نام نشریه.

۲۷-وجود یا عدم وجود واحد در مجموعه مشخص‌کننده‌های نام همایش.

۲۸-وجود یا عدم وجود واحد در مجموعه {"و"، "الی"، "تا"}.

۲۹-وجود یا عدم وجود واحد در مجموعه اسامی نویسندگان (استخراج شده با خزش از پایگاه سیویلیکا).

۳۰-امتیاز اختصاص داده شده به واحد (بین یک تا ۱۰) با توجه به فراوانی حضور واحد در عنوان نشریات (به دست آمده با خزش در پایگاه‌های https://www.civilica.com و http://isc.gov.ir).

۳۱-امتیاز اختصاص داده شده به واحد (بین یک تا ۱۰) با توجه به فراوانی حضور واحد در عنوان همایش‌ها (به دست آمده با خزش در پایگاه‌های https://www.civilica.com و http://isc.gov.ir).

۳۲-امتیاز اختصاص داده شده به واحد (بین یک تا ۱۰) با توجه به فراوانی حضور واحد در عنوان مقالات چاپ شده تا به امروز (به دست آمده با خزش در پایگاهhttps://www.civilica.com).

۳۳-باقی ماندن یا نماندن تنها عددی کمتر از ۱۰۰ پس از حذف مشخص‌کننده‌های دوره، شماره نشر، شماره جلد و شماره چاپ از ابتدای واحد (در صورت وجود).

۳۴-برچسب سومین واحد قبل از واحد فعلی. در صورت عدم وجود چنین واحدی در رشته مرجع از null استفاده می‌شود.

۳۵-برچسب دومین واحد قبل از واحد فعلی. در صورت عدم وجود چنین واحدی در رشته مرجع از null استفاده می‌شود.

۳۶-برچسب واحد قبل از واحد فعلی. در صورت عدم وجود چنین واحدی در رشته مرجع از null استفاده می‌شود.

۳۷- ۶۶ ویژگی‌های شماره ۴ تا ۳۳ بیان شده در بالا برای واحد بعدی در دنباله واحدهای رشته مرجع موردنظر. در صورتی که واحد موردنظر آخرین واحد در دنباله واحدها باشد از null برای تمام ویژگی‌ها استفاده خواهد شد.

دسته‌بند PerCitPars

- **آموزش:**

۱- هر رشته مرجع موجود در مجموعه آموزش به دنباله‌ای از واحدها تقسیم‌بندی شده و بردار ویژگی‌های متناظر با هر واحد محاسبه می‌شود. همانطور که ذکر شد، در این پژوهش از لغت به عنوان واحد استفاده شده است.

۲- دسته‌بند OSVM با استفاده از بردارهای ویژگی استخراج شده و برچسب‌های صحیح اختصاص‌یافته به هر واحد از هر رشته مرجع آموزش داده می‌شود.

- **تجزیه یک رشته مرجع جدید:**

۱- رشته مرجع به دنباله‌ای از واحدها تقسیم می‌شود.

۲- برای هر واحد موجود در رشته مرجع (به ترتیب از ابتدا تا انتها):

i. همه مولفه‌های بردار ویژگی متناظر با آن به جز مولفه‌های ۳۴، ۳۵ و ۳۶ محاسبه می‌شود.
ii. در صورتی که مقدار مولفه اول بردار ویژگی واحد مورد بررسی برابر با ۱ باشد، مقادیر مولفه‌های ۳۴، ۳۵ و ۳۶ آن برابر با null قرار داده می‌شود. در غیر این صورت، مقدار مولفه ۳۴ واحد فعلی برابر با مقدار مولفه ۳۵ بردار ویژگی واحد ماقبل، مقدار مولفه ۳۵ واحد فعلی برابر با مقدار مولفه ۳۶ بردار ویژگی واحد ماقبل و مقدار مولفه ۳۶ واحد برابر با برچسب اختصاص یافته به واحد ماقبل قرار می‌گیرد.
iii. با استفاده از دسته‌بند آموزش دیده، برچسب واحد فعلی به دست آمده و علاوه بر ذخیره، در بردارهای ویژگی واحدهای بعد مورد استفاده قرار می‌گیرد.
iv. با استفاده از مجموعه‌ای از قواعد اکتشافی، برچسب‌های اختصاص یافته به واحدها بهبود می‌یابند. برای دیدن لیست کامل قواعد به کار رفته به (پاک‌نیت و همکاران، ۱۳۹۷) مراجعه شود.
v. با کنار هم قرار دادن واحدهای با برچسب یکسان، فیلدهای مختلف یک رشته مرجع مشخص و معین می‌شوند.
vi. با توجه به نوع رشته مرجع، تنها فیلد متناظر با نوع (یعنی نام نشریه برای رشته‌های مرجع از نوع مقاله نشریه، نام همایش برای رشته‌های مرجع از نوع مقالات همایشی، نام دانشگاه برای رشته‌های مرجع از نوع پارسا و نام انتشارات برای رشته‌های مرجع از نوع کتاب) نگهداری شده و سه قلم اطلاعاتی دیگر از بین نام نشریه، نام همایش، نام دانشگاه، نام انتشارات حذف می‌گردد.
vii. با استفاده از الگوریتم SepAuths، مولفه نویسندگان یک رشته مرجع به لیستی از نام نویسندگان تجزیه می‌شود.

الگوریتم SepAuths (تجزیه مولفه نویسندگان)

برای دستیابی به نتایج مناسب در تجزیه مولفه نویسندگان، در ابتدا دسته‌بندی (که آن را AuthLableDet نامیده‌ایم) را طراحی کرده‌ایم که با ورودی مولفه نویسندگان که شامل دنباله‌ای از واحدها (کلمات) است، دنباله‌ای از

برچسب‌های "نام" و "نام خانوادگی" بازمی‌گرداند. در ادامه، این اطلاع اضافه به دست آمده را به کار گرفته و با استفاده از یک الگوریتمی ابتکاری، مولفه نویسندگان یک رشته مرجع به آرایه‌ای شامل نام خانوادگی و نام تک تک نویسندگان تجزیه می‌شود. در ادامه جزئیات دسته‌بند پیشنهادی برای اختصاص برچسب نام یا نام خانوادگی به هر واحد از مولفه نویسندگان یک رشته مرجع را بیان خواهیم کرد.

ویژگی‌های به کار رفته در دسته‌بند:

۱. برچسب دومین واحد قبل از واحد فعلی در دنباله‌ی واحدها.
۲. برچسب اولین واحد قبل از واحد فعلی در دنباله‌ی واحدها.
۳. موقعیت واحد در مولفه موردنظر.
۴. وجود واحد در مجموعه {"و"، "&"، "and"}.
۵. بزرگ بودن حرف اول واحد (در صورتی که واحد به زبان انگلیسی باشد).
۶. ختم یا عدم ختم واحد به کاراکتر '.'.
۷. ختم یا عدم ختم واحد به کاراکتر ','.
۸. ختم یا عدم ختم واحد به کاراکتر ';'.
۹. ختم یا عدم ختم واحد به کاراکتر '.'.
۱۰. ختم یا عدم ختم واحد به کاراکتر '-'.
۱۱. ختم یا عدم ختم واحد به کاراکتر '،'.
۱۲. ختم یا عدم ختم واحد به کاراکتر '؛'.
۱۳. وجود یا عدم وجود واحد در قالب نام مخفف نام یک نویسنده در زبان فارسی(مجموعه مخفف‌ها در بخش قبل ارائه شده‌اند).
۱۴. وجود یا عدم وجود واحد در قالب نام مخفف نام یک نویسنده در زبان انگلیسی.(مجموعه مخفف‌ها در بخش قبل ارائه شده‌اند).
۱۵. وجود واحد در مجموعه {"سید"، "سیده"، "السادات"}.
۱۶. وجود واحد در مجموعه {"etal"، "همکاران"، "دیگران"}.
۱۷. -۳۰: ویژگی‌های شماره ۴ تا ۱۶ ذکر شده در لیست فوق برای واحد بعدی در دنباله‌ی واحدها.

دسته‌بند EngTypeDet

- **آموزش:**

۱- مولفه‌های نویسندگان ورودی به دنباله‌ای از واحدها تقسیم شده و بردار ویژگی متناظر با هر واحد محاسبه می‌شود.
۲- دسته‌بند L2-SVM با استفاده از بردارهای ویژگی استخراج شده و برچسب‌های صحیح اختصاص‌یافته به هر واحد آموزش داده می‌شود.

- **تعیین نام یا نام خانوادگی بودن یک واحد:**

۱- مولفه ورودی به دنباله‌ای از واحدها تقسیم می‌شود.
۲- برای هر واحد موجود در دنباله (به ترتیب از ابتدا تا انتها):

i. همه مولفه‌های بردار ویژگی متناظر با آن به جز مولفه‌های ۱ و ۲ محاسبه می‌شود.
ii. در صورتی که مقدار مولفه سوم بردار ویژگی واحد مورد بررسی برابر با ۱ باشد، مقادیر مولفه‌های اول و دوم آن برابر با null قرار داده می‌شود. در غیر این صورت، مقدار مولفه اول واحد فعلی برابر با مقدار مولفه دوم بردار ویژگی واحد ماقبل و مقدار مولفه دوم واحد فعلی برابر با برابر با برچسب اختصاص یافته به واحد ماقبل قرار می‌گیرد.
iii. با استفاده از دسته‌بند آموزش دیده، برچسب واحد فعلی به دست آمده و علاوه بر ذخیره، در بردارهای ویژگی واحدهای بعد مورد استفاده قرار می‌گیرد.

ارزیابی دسته‌بند AuthLableDet

برای ارزیابی دسته‌بند AuthLableDet، از مولفه نویسندگان ۱۵۰ رشته مرجع و روش اعتبارسنجی متقابل ۱۰-سطحی استفاده شده است. نتایج پیاده‌سازی دسته‌بند پیشنهادی روی مجموعه داده موردنظر نشان می‌دهد که میانگین دقت، فراخوانی و F1 در روش ارائه شده برابر با ۹۲٪ است. این نتایج به صورت جزئی و برای دو نوع نام و نام خانوادگی به صورت مجزا در جدول شماره ۶ گزارش شده است.

جدول ۷. نتایج ارزیابی دسته‌بند پیشنهادی برای تعیین نام یا نام خانوادگی بودن یک واحد از مولفه نویسندگان یک سند علمی.

برچسب	دقت	فراخوانی	F1	تعداد
نام	۰,۹۴	۰,۹۲	۰,۹۳	۲۷۴
نام خانوادگی	۰,۹۲	۰,۹۴	۰,۹۳	۲۷۴
مجموع	۰,۹۳	۰,۹۳	۰,۹۳	۵۴۸

در ادامه از یک الگوریتم ابتکاری به نام SepAuths برای تجزیه مولفه نویسندگان یک رشته مرجع به دنباله اسامی نویسندگان آن به صورت مجزا استفاده می‌شود. الگوریتم SepAuths یک الگوریتم بازگشتی است که برای استخراج اسامی نویسندگان مجزا از طول دنباله‌ی واحدهای مولفه نویسندگان یک رشته مرجع و دنباله‌ی برچسب‌های متناظر آن‌ها (به دست آمده با استفاده از دسته‌بند AuthLabelDet) استفاده می‌کند. جزئیات این الگوریتم در کد ارائه شده در بخش ضمیمه قابل مشاهده است.

ارزیابی الگوریتم SepAuths: با انجام آزمایش روی مجموعه داده به کار گرفته شده برای آموزش دسته‌بند AuthLabelDet مشخص شد که الگوریتم SepAuths در ۹۲٪ مواقع، نویسندگان را به درستی تشخیص می‌دهد.

ارزیابی دسته‌بند PerCitPars

ارزیابی دسته‌بند ارائه شده برای تجزیه رشته‌های مرجع به زبان فارسی با استفاده از مجموعه داده طراحی شده در (پاک‌نیت و همکاران ۱۳۹۷) و روش اعتبارسنجی متقابل ۱۰-سطحی انجام شده است. مجموعه داده موردنظر شامل ۹۰۰ رشته مرجع می‌باشد که به هر واحد در هر رشته از این مجموعه داده یک برچسب از مجموعه برچسب‌های {نام نویسنده، عنوان، سال، نشریه، دوره/شماره، شماره صفحه، انتشارات، آدرس،یادداشت، موسسه، همایش، ماه/فصل} اختصاص یافته است. نتایج پیاده‌سازی دسته‌بند پیشنهادی روی مجموعه داده‌های موردنظر نشان می‌دهد که میانگین دقت، فراخوانی و F1 در روش ارائه شده برابر با ۹۲٪ است. این نتایج به صورت جزئی و برای هر برچسب به صورت مجزا در جدول شماره ۷ گزارش شده‌اند.

جدول ۸. نتایج ارزیابی دسته‌بند پیشنهادی برای تجزیه رشته‌های مرجع به زبان فارسی به مولفه‌های سازنده آن‌ها

برچسب	دقت	فراخوانی	F1	تعداد
نام نویسنده	۰,۹۹	۰,۹۹	۰,۹۹	۴۱۷۳
عنوان	۰,۹۸	۰,۹۹	۰,۹۹	۱۰۲۴۴
سال	۰,۹۹	۰,۹۹	۰,۹۹	۱۰۴۶
نشریه	۰,۹۷	۰,۹۱	۰,۹۴	۹۷۳
دوره/شماره	۰,۸۷	۰,۹۱	۰,۸۹	۴۲۳
شماره صفحه	۰,۸۹	۰,۸۸	۰,۸۹	۲۷۴

انتشارات	۰,۹۳	۰,۸۶	۰,۸۹	۸۶۶
آدرس	۰,۷۷	۰,۷۴	۰,۷۵	۷۹۳
غیره	۰,۹۲	۰,۹۱	۰,۹۱	۲۲۵۶
موسسه	۰,۸۰	۰,۸۱	۰,۸۱	۸۰۰
همایش	۰,۹۲	۰,۹۳	۰,۹۲	۱۹۰۱
شماره چاپ	۰,۷۴	۰,۷۵	۰,۷۵	۸۴
ماه/فصل	۰,۹۸	۰,۹۶	۰,۹۷	۱۶۴
مجموع	۰,۹۵	۰,۹۵	۰,۹۵	۲۳۹۹۷

نمونه‌هایی از نتایج به دست آمده با استفاده از دسته‌بند PerCitPars

در ادامه و در جدول ۸، نمونه‌هایی از نتایج حاصل از اعمال دسته‌بند PerCitPars روی تعدادی رشته مرجع بیان خواهد شد. نمونه‌های ذکر شده شامل ارجاعاتی به مقالات چاپ شده در نشریه و همایش، پایان‌نامه دانشگاهی و کتاب است. علاوه‌براین، برای خوانایی بیشتر، از نمادگذاری‌های زیر برای برچسب‌ها استفاده شده است:

A: نام نویسنده	T: عنوان
Y: سال	J: نشریه
N: شماره	PP: شماره صفحه
Pub: انتشارات	Add: آدرس
No: یادداشت	Sch: موسسه
C: همایش	M/S: ماه/فصل

در این جدول، "پس‌زمینه رنگی" به معنای اختصاص اشتباه برچسب به قسمت با پس‌زمینه رنگی از متن و "خط کشیده شده در زیر متن" به معنای عدم احتصاص برچسب موردنظر به قسمت مشخص شده است.

جدول ۹. نمونه‌هایی از رشته‌های مرجع تجزیه شده با دسته‌بند PerCitPars. ستون اول، رشته دریافتی ورودی است و ستون دوم مولفه‌های مستخرج شده از آن‌ها.

۱۶-نوغانی، محسن.، اصغرپور ماسوله، احمدرضا، صفا، شیما. و کرمانی، مهدی. ۱۳۸۸. کیفیت زندگی شهروندان و رابطه‌ی ان با سرمایه اجتماعی در شهر مشهد، مجله علوم اجتماعی دانشکده ادبیات و علوم انسانی، دانشگاه	نوغانی، محسن (A1) اصغرپور ماسوله، احمدرضا (A2) صفا، شیما (A3) کرمانی، مهدی (A4)

فردوسی مشهد، بهار و تابستان.	(Y) ۱۳۸۸ کیفیت زندگی شهروندان و رابطه‌ی ان با سرمایه اجتماعی در شهر مشهد (T) مجله علوم اجتماعی دانشکده ادبیات و علوم انسانی دانشگاه فردوسی مشهد (J) دانشگاه فردوسی مشهد (Add) بهار و تابستان (M/S).
۲.اسکندری، سعیده و دیگران. ۱۳۹۲. ارزیابی کارایی مدل دانگ برای تعیین قابلیت خطر اتش‌سوزی در جنگل‌های زرین‌اباد نکا، استان مازندران، فصل‌نامه تحقیقات جنگل و صنوبر ایران، جلد ۲۱، شماره ۳، صص ۴۵۱-۴۳۹.	اسکندری سعیده (A1) (Y) ۱۳۹۲ ارزیابی کارایی مدل دانگ برای تعیین قابلیت خطر اتش سوزی در جنگل‌های زرین‌اباد نکا، استان مازندران (T) فصل‌نامه تحقیقات جنگل و صنوبر ایران (J) (N) (۳)۲۱ (PP) ۴۵۱-۴۳۹
نوروزی، مهناز، پاک نیت، نصراله، اسلامی، زیبا. رمزگذاری کلید عمومی با قابلیت جستجوی کلیدواژه: ارایه یک ساخت کلی امن در برابر حملات حدس کلیدواژه برخط و غیربرخط، چهاردهمین کنفرانس بین المللی انجمن رمز ایران، شهریور ۱۳۹۶، شیراز، ایران	نوروزی مهناز، پاک نیت نصراله (A1) اسلامی، زیبا (A2) رمزگذاری کلید عمومی با قابلیت جستجوی کلیدواژه: ارایه یک ساخت کلی امن در برابر حملات حدس کلیدواژه برخط و غیربرخط (T) چهاردهمین کنفرانس بین‌المللی انجمن رمز ایران (C) شهریور (M/S) (Y) ۱۳۹۶ شیراز ایران (Add)
اصفهانی، محمدمهدی. ۱۳۸۴. ایین تندرستی. چاپ اول. تهران : اداره ارتباطات و اموزش سلامت معاونت سلامت وزارت بهداشت، درمان و اموزش پزشکی.	اصفهانی، محمدمهدی (A1) (Y) ۱۳۸۴ ایین تندرستی (T) اول (No) تهران (Add) اداره ارتباطات و اموزش سلامت معاونت سلامت وزارت بهداشت، درمان و اموزش پزشکی (Pub)
[۱۲] عدالت‌پناه،ف. و رضازاده،پ، "پیش‌بینی پارامترهای موج با استفاده از مدل SWAN"، دوازدهمین کنفرانس	عدالت‌پناه، ف (A1) رضازاده، پ (A2)

دینامیک شاره‌ها، دانشگاه نوشیروانی بابل، ۱۳۸۸.	پیش‌بینی پارامترهای موج با استفاده از مدل SWAN (T) دوازدهمین کنفرانس دینامیک شاره‌ها (C) دانشگاه نوشیروانی بابل (Add) ۱۳۸۸ (Y)
گایینی، عباسعلی، علیدوست قهفرخی، ابراهیم، احمدی، علی. (۱۳۸۸)، «تاثیر مصرف کوتاه‌مدت مکمل کراتین بر عملکردهای سرعتی و قدرت عضلانی کشتی‌گیران»، علوم زیستی و ورزشی، شماره ۳: ص ۷۷ تا ۹۲.	گایینی عباسعلی، علیدوست (A1) قهفرخی ابراهیم ، احمدی علی (A2) ۱۳۸۸ (Y) تاثیر مصرف کوتاه‌مدت مکمل کراتین بر عملکردهای سرعتی و قدرت عضلانی کشتی‌گیران علوم زیستی و ورزشی (T) ۳(N) ۷۷ تا ۹۲ (PP).
[۳۱] ساسانیان، سعید؛ بررسی اثر تغییرات pH شیرابه بر خصوصیات مکانیکی پوششهای رسی(CCLها)، پایان نامه کارشناسی ارشد، دانشکده عمران، دانشگاه صنعتی شریف،۱۳۸۳	ساسانیان، سعید (A1) بررسی اثر تغییرات PH شیرابه بر خصوصیات مکانیکی پوششهای رسی (CCLها) (T) دانشکده عمران دانشگاه صنعتی شریف (Sch) ۱۳۸۳ (Y)
مارکس، کارل. ۱۳۸۸. سرمایه : نقدی بر اقتصاد سیاسی. ترجمه : حسن مرتضوی، تهران : نشر اگاه.	مارکس، کارل (A1) ۱۳۸۸ (Y) سرمایه نقدی بر اقتصاد سیاسی (T) تهران نشر اگاه (Pub)

٤-٥-٢. تجزیه رشته‌های مرجع در زبان انگلیسی

در این بخش، از روشی مشابه با روش ارائه شده در بخش قبل برای تجزیه یک رشته مرجع به زبان انگلیسی به مولفه‌های سازنده ارائه می‌کنیم. در دسته‌بند ارائه شده در این بخش نیز، یک رشته مرجع به مجموعه‌ای از لغات به عنوان واحدها تقسیم می‌شود و دسته‌بندی در سطح لغت انجام می‌شود. لازم به ذکر است که مجموعه‌های مورد استفاده در این بخش برابر با مجموعه‌های مورد استفاده در بخش ۴-۴-۲ هستند.

ویژگی‌های به کار رفته در دسته‌بند:

۱- موقعیت واحد در رشته مرجع.

۲- وجود یا عدم وجود واحد در مجموعه {"in"، ":in"}.

۳- وجود یا عدم وجود واحد در مجموعه {"eds"}.

۴- وجود یا عدم وجود یکی از علائم نقطه‌گذاری {","، "."، ";"، "\"، "»"، "«"، "?"، "("، ")"} پس از واحد.

۵- بزرگ یا کوچک بودن حرف اول واحد.

۶- ختم یا عدم ختم واحد به کاراکتر ":".

۷- محصور بودن یا نبودن واحد به "(" و ")".

۸- وجود یا عدم وجود واحد در مجموعه مشخص‌کننده‌های شماره صفحه.

۹- وجود یا عدم وجود واحد در قالب شماره صفحه.

۱۰- عدد یا غیر عدد بودن واحد.

۱۱- تعداد واحدهای عددی مشاهده شده قبل از واحد فعلی.

۱۲- بودن یا نبودن واحد در قالب سال.

۱۳- بودن یا نبودن واحد در قالب نام یک نویسنده (مخفف نام نویسنده). بدین منظور وجود یا عدم وجود واحد در مجموعه مشخص‌کننده‌های مخفف یک نام انگلیسی (ارائه شده در بخش قبل) بررسی می‌گردد.

۱۴- وجود یا عدم وجود کاراکتری غیر از کاراکترهای متناظر با حروف انگلیسی در واحد.

۱۵- وجود یا عدم وجود همزمان حروف الفبای انگلیسی و عدد در واحد.

۱۵-وجود یا عدم وجود کاراکتری غیر حروف الفبای انگلیسی و عدد در واحد.

۱۶-وجود یا عدم وجود واحد در مجموعه مشخص‌کننده‌های دوره.

۱۷-وجود یا عدم وجود واحد در مجموعه مشخص‌کننده‌های شماره.

۱۸-وجود یا عدم وجود واحد در مجموعه مشخص‌کننده شماره چاپ.

۱۹-وجود یا عدم وجود واحد در مجموعه نام ماه‌ها و فصل‌های یک سال.

۲۰-وجود یا عدم وجود واحد در مجموعه {"a""، "an"، "and"، "or"، "the"، "of"، "on"}.

۲۱- وجود یا عدم وجود واحد در مجموعه اعداد نوشته شده. بدین منظور از نگارش انگلیسی اعداد کمتر از صد استفاده شده است.

۲۲-وجود یا عدم وجود واحد در مجموعه {"etal"}.

۲۳-وجود یا عدم وجود واحد در مجموعه مشخص‌کننده‌های نام دانشگاه.

۲۴-وجود یا عدم وجود واحد در مجموعه مشخص‌کننده‌های پارسا.

۲۵-وجود یا عدم وجود واحد در مجموعه مشخص‌کننده‌های انتشارات.

۲۶-وجود یا عدم وجود واحد در مجموعه لغات به دست آمده از لیست نام انتشارات‌های مهم. برای نام انتشارات‌های مهم از مجموعه انتشارات با رتبه A و رتبه B گزارش شده در آدرس https://research.usp.ac.fj/wp-content/uploads/2013/09/2016-Ranking-of-Academic-Publishers-v1.0.pdf استفاده شده است.

۲۷-وجود یا عدم وجود واحد در لیست اسامی شهرهای مهم.

۲۸-وجود یا عدم وجود واحد در مجموعه مشخص‌کننده‌های نام نشریه.

۲۹-وجود یا عدم وجود واحد در مجموعه مشخص‌کننده‌های نام همایش.

۳۰-امتیاز اختصاص داده شده به واحد (بین یک تا ۱۰) با توجه به فراوانی حضور واحد در عنوان نشریات (به دست آمده از www.scopus.com).

۳۱-امتیاز اختصاص داده شده به واحد (بین یک تا ۱۰) با توجه به فراوانی حضور واحد در عنوان همایش‌ها (به دست آمده از www.scopus.com).

۳۲-برچسب سومین واحد قبل از واحد فعلی. در صورت عدم وجود چنین واحدی در رشته مرجع از null استفاده می‌شود.

۳۳-برچسب دومین واحد قبل از واحد فعلی. در صورت عدم وجود چنین واحدی در رشته مرجع از null استفاده می‌شود.

۳۴-برچسب واحد قبل از واحد فعلی. در صورت عدم وجود چنین واحدی در رشته مرجع از null استفاده می‌شود.

۳۵- تا 64 ویژگی‌های شماره ۲ تا ۳۱ بیان شده در بالا برای واحد بعدی در دنباله واحدهای رشته مرجع موردنظر. در صورتی که واحد موردنظر آخرین واحد در دنباله واحدها باشد از null برای تمام ویژگی‌ها استفاده خواهد شد.

دسته‌بند EngCitPars

- **آموزش:**

۱- هر رشته مرجع موجود در مجموعه آموزش به دنباله‌ای از واحدها تقسیم‌بندی شده و بردار ویژگی‌های متناظر با هر واحد محاسبه می‌شود. همانطور که ذکر شد، در این پژوهش از لغت به عنوان واحد استفاده شده است.

۲- دسته‌بند OSVM با استفاده از بردارهای ویژگی استخراج شده و برچسب‌های صحیح اختصاص‌یافته به هر واحد از هر رشته مرجع آموزش داده می‌شود.

- **تجزیه یک رشته مرجع جدید:**

۱- رشته مرجع به دنباله‌ای از واحدها تقسیم می‌شود.

۲- برای هر واحد موجود در رشته مرجع (به ترتیب از ابتدا تا انتها):

i. همه مولفه‌های بردار ویژگی متناظر با آن به جز مولفه‌های ۳۲، ۳۳ و ۳۴ محاسبه می‌شود.

ii. در صورتی که مقدار مولفه اول بردار ویژگی واحد مورد بررسی برابر با ۱ باشد، مقادیر مولفه‌های ۳۲، ۳۳ و ۳۴ آن برابر با null قرار داده می‌شود. در غیر این صورت، مقدار مولفه ۳۲ واحد فعلی برابر با مقدار مولفه ۳۳ بردار ویژگی واحد ماقبل، مقدار مولفه ۳۳ واحد فعلی برابر با مقدار مولفه ۳۴ بردار ویژگی واحد ماقبل و مقدار مولفه ۳۴ واحد برابر با برچسب اختصاص یافته به واحد ماقبل قرار می‌گیرد.

iii. با استفاده از دسته‌بند آموزش دیده، برچسب واحد فعلی به دست آمده و علاوه بر ذخیره، در بردارهای ویژگی واحدهای بعد مورد استفاده قرار می‌گیرد.

iv. با استفاده از مجموعه‌ای از قواعد اکتشافی، برچسب‌های اختصاص یافته به واحدها بهبود می‌یابند.

v. با کنار هم قرار دادن واحدهای با برچسب یکسان، فیلدهای مختلف یک رشته مرجع مشخص و معین می‌شوند.

vi. با استفاده از الگوریتم SepAuths، مولفه نویسندگان یک رشته مرجع به لیستی از نام نویسندگان تجزیه می‌شود.

- **آموزش:**

۱- هر رشته مرجع موجود در مجموعه آموزش به دنباله‌ای از واحدها تقسیم‌بندی شده و بردار ویژگی‌های متناظر با هر واحد محاسبه می‌شود. همانطور که ذکر شد، در این پژوهش از لغت به عنوان واحد استفاده شده است.

۲- دسته‌بند OSVM با استفاده از بردارهای ویژگی استخراج شده و برچسب‌های صحیح اختصاص‌یافته به هر واحد از هر رشته مرجع آموزش داده می‌شود.

- **تجزیه یک رشته مرجع جدید:**

۱- رشته مرجع به دنباله‌ای از واحدها تقسیم می‌شود.

۲- برای هر واحد موجود در رشته مرجع (به ترتیب از ابتدا تا انتها):

i. بردار ویژگی متناظر با آن محاسبه می‌شود.

ii. با استفاده از دسته‌بند آموزش دیده، برچسب واحد محاسبه می‌شود.

iii. با کنار هم قرار دادن واحدهای با برچسب یکسان، فیلدهای مختلف یک رشته مرجع مشخص و معین می‌شوند.

iv. با توجه به نوع رشته مرجع، تنها فیلد متناظر با نوع (یعنی نام نشریه برای رشته‌های مرجع از نوع مقاله نشریه، نام همایش برای رشته‌های مرجع از نوع مقالات همایشی، نام دانشگاه برای رشته‌های مرجع از نوع پارسا و نام انتشارات برای رشته‌های مرجع از نوع کتاب) نگهداری شده و سه قلم اطلاعاتی دیگر از بین نام نشریه، نام همایش، نام دانشگاه، نام انتشارات حذف می‌گردد.

v. با استفاده از الگوریتم SepAuths (بیان شده در قسمت قبل)، مولفه نویسندگان یک رشته مرجع به لیستی از نام تک تک نویسندگان تجزیه می‌شود.

ارزیابی دسته‌بند EngCitPars

ارزیابی دسته‌بند ارائه شده برای تجزیه رشته‌های مرجع به زبان انگلیسی با استفاده از مجموعه داده طراحی شده در این پژوهش و روش اعتبارسنجی متقابل ۱۰-سطحی انجام می‌شود. مجموعه داده موردنظر شامل ۲۹۲ رشته مرجع (۵۱ کتاب، ۵۹ مقاله چاپ شده در نشریه، ۱۷۹ مقاله ارائه شده در همایش و ۳ پارسا) می‌باشد که به هر واحد در هر رشته از این مجموعه داده یک برچسب از مجموعه برچسب‌های {نام نویسنده، عنوان، سال، نشریه، دوره/شماره، شماره صفحه، انتشارات، آدرس، یادداشت، موسسه، همایش، ماه/فصل} اختصاص یافته است. نتایج پیاده‌سازی دسته‌بند پیشنهادی روی مجموعه داده‌های موردنظر نشان می‌دهد که میانگین دقت، فراخوانی و F1 در روش ارائه شده برابر با ۸۷٪ است. این نتایج به صورت جزئی و برای هر برچسب به صورت مجزا در جدول ۹ گزارش شده‌اند.

جدول ۱۰. نتایج ارزیابی روش دسته‌بند پیشنهادی برای تجزیه رشته‌های مرجع به زبان انگلیسی به مولفه‌های آن‌ها

برچسب	دقت	فراخوانی	F1	تعداد
نام نویسنده	۰,۹۶	۰,۹۷	۰,۹۶	۵۶۲۷
عنوان	۰,۹۳	۰,۹۸	۰,۹۵	۹۶۹۰
سال	۰,۹۸	۰,۹۹	۰,۹۹	۱۰۳۰
نشریه	۰,۹۴	۰,۷۵	۰,۸۳	۱۰۱۸
شماره	۰,۹۳	۰,۹۴	۰,۹۴	۳۸۵
شماره صفحه	۰,۹۶	۰,۹۶	۰,۹۶	۴۴۱
انتشارات	۰,۷۹	۰,۶۶	۰,۷۲	۸۱۲
آدرس	۰,۶۸	۰,۵۹	۰,۶۳	۶۶۰
غیره	۰,۸۶	۰,۷۴	۰,۸۰	۱۹۹۵
موسسه	۰,۸۲	۰,۶۲	۰,۷۱	۱۰۶۸
همایش	۰,۷۷	۰,۹۲	۰,۸۴	۲۱۰۴
شماره چاپ	۰,۹۳	۰,۸۷	۰,۹۰	۳۱
ماه/فصل	۰,۷۵	۰,۸۴	۰,۷۹	۴۹
مجموع	۰,۹۰	۰,۹۰	۰,۹۰	۲۴۹۱۰

نمونه‌هایی از نتایج به دست آمده با استفاده از دسته‌بند EngCitPars

در ادامه و در جدول ۱۰، نمونه‌هایی از نتایج حاصل از اعمال دسته‌بند EngCitPars روی تعدادی رشته مرجع بیان خواهد شد. نمونه‌های ذکر شده شامل ارجاعاتی به مقالات چاپ شده در نشریه و همایش، پایان‌نامه دانشگاهی و کتاب است. در این جدول، "پس‌زمینه رنگی" به معنای اختصاص اشتباه برچسب به قسمت با پس‌زمینه رنگی از متن و "خط کشیده شده در زیر متن" به معنای عدم احتصاص برچسب موردنظر به قسمت مشخص شده است.

جدول ۱۱. نمونه‌هایی از رشته‌های مرجع تجزیه شده با دسته‌بند EngCitPars. ستون اول، رشته دریافتی ورودی است و ستون دوم مولفه‌های مستخرج شده از آن‌ها.

Johnson, M W (A1) Christensen, C C (A2) Kagermann, H (A3) 2008 (Y) reinventing your business model harvard business review (T) harvard business review (J) 86(12) (N) 50-59 (PP)	Johnson, M. W., Christensen, C. C., Kagermann, H. (2008). Reinventing your business model. Harvard Business Review 86(12): 50-59.
Anderson, A (A1) Ackerman Anderson, L (A2) 2010 (Y) beyond change management: how to achieve breakthrough results through conscious leadership san francisco, united (T) Imperint of Wiley (Pub)	Anderson, A., Ackerman Anderson, L. (2010) Beyond Change Management: How to Achieve Breakthrough Results through Conscious Leadership. San Francisco, United States : An Imperint of Wiley.
Shenhar, A (A1) strategic project management: the new framework (T) the portland international conference on management (C)	۷۳. Shenhar A. Strategic project management: the new framework. In:Proceedings of the Portland International Conference on Management
Farkas, A (A1) Salánki, J (A2) Specziár, A (A3) 2003 (Y) age-and site-specific patterns of heavy metals in the organs of freshwater fish abrams bram populating a low-contaminated site (T) water research (J) 37(5) (N) 959-964 (PP)	Farkas, A., Salánki, J., & Specziár, A. (2003). Age-and Site-Specific Patterns of Heavy Metals in the Organs of Freshwater Fish Abrams Bram L. Populating a low-contaminated Site. Water research, 37(5), 959-964 .
McQuarrie, N (A1) Vanhinsbergen, D J (A2) Geology, J J (A3) 2013 (Y) retrodeforming the arabia-eurasia collision zone: age of collision versus magnitude of continental subduction: (T) Geology (J) 41 (N) 315-318 (PP)	McQuarrie, N., and van Hinsbergen, D.J.J., 2013, Retrodeforming the Arabia-Eurasia collision zone: Age of collision versus magnitude of continental subduction: Geology, v. 41, p. 315–318.
Balogun, J (A1)	Balogun, J., & Hailey, V.H., Johnson, G., and

Hailey, V H (A2) Johnson, G (A3) Scholes, K (A4) 1999 (Y) exploring strategic change (T) great britain prentice hall europe (Pub)	Scholes, K. (1999). Exploring Strategic Change (1st ed.). Great Britain: Prentice Hall Europe .
Collymore, P (A1) Erskine, R (A2) 1994 (Y) the architecture of ralph erskine (T) london (Add) academy editions ltd press (P)	□ Collymore, P., & Erskine, R.,1994:The Architecture of Ralph Erskine. London:Academy Editions Ltd Press .

٤-٦- الگوریتم‌های تطبیق

در این بخش، الگوریتم‌های به کار رفته برای تطبیق مولفه‌های مختلف رشته‌های مرجع ارائه می‌شوند. این الگوریتم‌ها بر اساس معیار فاصله ویرایش لون‌اشتاین هستند. همانطور که در بخش ۲-۳ بیان شد، محاسبه فاصله لون‌اشتاین دو رشته دارای پیچیدگی محاسباتی $O(mn)$ است که m و n طول دو رشته مورد مقایسه هستند. به نظر می‌رسد با توجه به تعداد زیاد مقایسه موردانتظار برای ساخت شبکه استنادی پارساهای موجود در پایگاه داده گنج، تنها استفاده از این الگوریتم کارایی مناسبی نخواهد داشت و باید با استفاده از مکانیزم‌هایی کارایی روش را ارتقا داد. با توجه به اینکه در این پژوهش به دنبال رشته‌های به اندازه کافی نزدیک به یک رشته در مقایسات هستیم، یک رویکرد این است که محاسبه فاصله ویرایش دو رشته را تا زمانی که فاصله آن‌ها کمتر از مقداری مشخص است ادامه دهیم. به عبارت دیگر، در صورتی که متوجه شدیم که فاصله ویرایش دو رشته مرجع از حدی بیشتر شد، دو رشته را نامطابق در نظر گرفته و از محاسبه فاصله ویرایش دقیق بین دو رشته صرف‌نظر کنیم[۱]. یک رویکرد دیگر این است که در ابتدا با استفاده از یک معیار شباهت کاراتر، مجموعه‌ای از رشته‌های احتمالا مطابق که بسیار کوچک‌تر از مجموعه اصلی است را استخراج کرده و در ادامه با استفاده از معیار فاصله ویرایش لون‌اشتاین، شبیه‌ترین رشته به رشته ورودی مشخص شود. در این پژوهش از رویکرد دوم استفاده کرده و ابتدا مجموعه n-تایی‌های دو رشته با استفاده از معیار ژاکارد مورد مقایسه قرار گرفته و در ادامه، در صورتی که درصد مشابهت دو مجموعه از مقداری آستانه‌ای بالاتر باشد، با استفاده از معیار فاصله ویرایش لون‌اشتاین میزان مشابهت دقیق دو رشته بررسی می‌شود.

٤-٦-١. تطبیق نام نویسندگان

[۱] https://en.wikipedia.org/wiki/Levenshtein_distance

همانطور که قبلا نیز ذکر شد، نگارش نام نویسندگان در مراجع ممکن است به شکل‌های متفاوت باشد. با توجه به الگوریتم ارائه شده برای جداسازی نام نویسندگان در بخش ۴,۵,۱، پس از تجزیه یک رشته مرجع، اسامی نویسندگان آن رشته مرجع را به صورت مجزا در اختیار داریم. علاوه براین، خروجی آماده شده توسط آن الگوریتم بخش متناظر با نام و بخش متناظر با نام خانوادگی نویسندگان را نیز مشخص می‌کرد. با توجه به این توضیحات، برای تطبیق نام نویسندگان به صورت زیر عمل می‌کنیم:

الگوریتم AuthMatch

۱- تمام حروف انگلیسی نام نویسنده به حروف کوچک تبدیل می‌شوند.
۲- حرف اول مولفه نام نویسنده به انتهای مولفه نام خانوادگی او چسبانده شده و هر کاراکتری از این رشته که در مجموعه حروف الفبا قرار نداشته باشد، حذف می‌شود.
۳- در ادامه به جستجوی دقیق رشته حاصل در جدول نام نویسندگان پرداخته و در صورت وجود یک تطبیق، شناسه متناظر بازگردانده می‌شود.
۴- در صورت عدم وجود تطابق، null بازگردانده می‌شود.

نمونه‌هایی از نتایج به دست آمده با استفاده از الگوریتم AuthMatch

در ادامه و در جدول ۱۱، نمونه‌هایی از نتایج حاصل از اعمال دسته‌بند AuthMatch روی تعدادی نام نویسنده ورودی ارائه می‌شود.

جدول ۱۲. نمونه‌های از نام‌های یکسان تشخیص داده شده توسط الگوریتم AuthMatch.

م بیات	مقصود بیات
لیلا حیدری نسب	لیلا حیدری‌نسب
محمدرضا شعیری	محمدرضا شعیری
م تقوایی	مسعود تقوایی
ع احمدی	علی محمد احمدی
ع عسگری	علی عسگری

م بیات	مریم بیات
محسن احدنژادروشتی	محسن احدنژاد روشتی
و فرزاد	ولی‌اله فرزاد
م حسن زاده	محمد حسن‌زاده
G Smith	Gill Wyness Smith
Ahola, K	Ahola, Kisoke
Cook, L.	Cook, L. C.
Maslach,Charles	Maslach,C
Tardy, C. H.	Tardy, Charles H

٤-٦-٢. تطبیق نام محل‌ها

با توجه به وجود برخی ایرادات، مانند ناقص نوشتن نام محل‌ها، الگوریتم زیر برای جستجوی نام یک محل در جدول نام محل‌ها پیشنهاد می‌گردد:

الگوریتم VenueMatch

۱- تمام حروف انگلیسی نام محل به حروف کوچک تبدیل می‌شوند.

۲- مجموعه n-تایی‌های موجود در نام محل به دست آمده و با توجه به معیار ژاکارد با مجموعه n-تایی‌های هر نام محل ذخیره شده در جدول محل‌ها (با نوع یکسان با رشته مرجع دربرگیرنده محل (مقاله نشریه، مقاله همایش، کتاب، پارسا) مقایسه می‌شود. فرض کنید PM مجموعه شناسه‌های متناظر با اعضایی از جدول محل‌ها باشد که میزان مشابهت به دست آمده به ازای آن‌ها حداقل t_{ven}^{1} باشد. در آزمایشات محدود صورت گرفته، نتایج مناسبی با استفاده از n=3 و $t_{tit}^{1} = 0.5$ به دست آمده است. تعیین مقدار دقیق‌تر برای این پارامترها در گام بعدی صورت خواهد گرفت.

۳- به ازای هر شناسه موجود در PM:

a. با استفاده از الگوریتم لون‌اشتاین، فاصله بین نام متناظر با این شناسه و رشته X محاسبه می‌شود.

۴- فرض کنیم ID شناسه‌ای باشد که در گام فوق کمترین مقدار به ازای آن به دست آمده است. در صورتی که میزان مشابهت X و نام محل متناظر با ID حداقل برابر با t_{ven}^{2} باشد، شناسه ID به عنوان یک تطابق بازگردانده می‌شود. در آزمایشات صورت گرفته، از ۰٫۵ به عنوان مقدار این پارامتر استفاده شده است.

۵- در صورتی که عضوی از مجموعه PM وجود داشته باشد که میزان مشابهت آن با محل مورد جستجو و با استفاده از ضریب هم‌پوشانی در الگوریتم n-تایی‌ها حداقل t_{ven}^{3} باشد، شناسه متناظر با نام این محل خروجی داده می‌شود.

در موارد بسیاری مشاهده شده که نام محل‌ها در مراجع به صورت کامل ذکر نشده یا با اطلاعاتی اضافه نوشته شده است. به عنوان چند مثال در این راستا می‌توان به موارد زیر اشاره کرد: "دانشگاه امام صادق (ع)" و "دانشگاه امام صادق"، "مجله جغرافیا و توسعه" و "جغرافیا و توسعه"، "دانشکده علوم تربیتی دانشگاه شهید بهشتی" و "دانشگاه شهید بهشتی" و ... . معیار ارائه شده در این مرحله برای در نظر گرفتن چنین مواردی استفاده شده است. لازم به ذکر است که در آزمایشات انجام شده از مقدار ۰,۹۵ به عنوان پارامتر t_{ven}^{3} استفاده شده است.

۶- در صورت عدم برقراری هیچ یک از روابط فوق، null بازگردانده خواهد شد.

نمونه‌هایی از نتایج به دست آمده با استفاده از الگوریتم VenueMatch

در ادامه و در جدول ۱۲، نمونه‌هایی از نتایج حاصل از اعمال دسته‌بند VenueMatch روی تعدادی نام محل ورودی ارائه می‌شود. نمونه‌های موجود در این جدول، به صورت دستی و با شبیه‌سازی خطاهای ممکن در استخراج مولفه‌های یک رشته مرجع ایجاد شده‌اند.

جدول ۱۳. نمونه‌هایی از تطابق‌های تشخیص داده شده با استفاده از الگوریتم VenueMatch. پس‌زمینه موارد اشتباه تشخیص داده شده رنگی شده است.

دانشگاه علامه طباطبایی	دانشگاه علامه طباطبایی(ره) تهران
دانشگاه علامه طباطبایی(ره)	دانشگاه علامه طباطبایی(ره) تهران
دانشگاه علامه طباطبایی(ره)	دانشگاه علامه طباطبایی
مجله¬ی دست¬آورد¬های روان¬شناختی	مجله دست آوردهای روانشناختی
انتشارات علمی و فرهنگی	انتشارات علمی
انتشارات دانشگاه تهران	انتشارات دانشگاه
انتشارات دانشگاه مشهد	انتشارات دانشگاه فردوسی مشهد
مجله دانشکده ادبیات و علوم انسانی تهران	مجله دانشکده ادبیات و علوم انسانی دانشگاه تهران
Personality & individual differences	Personality and Individual Differences
Consulting Psychologists Press	Consulting psychologists Press Inc

Journal of College Student Development	Journal ofCollege and Student Development
European Journal of Personality	European Journal of Personality, Eur. J. Pers

٤-٦-٣. تطبیق عنوان‌ها

به دلایل بسیاری من جمله اشتباهات نگارشی و خطا در تجزیه رشته‌های مرجع، مقایسه دقیق عناوین ممکن است منجر به نتایجی نامناسب شود. با توجه به این مساله، الگوریتم زیر برای جستجوی یک عنوان در ستون عناوین جدول اسناد پیشنهاد می‌گردد:

الگوریتم TitleMatch

۱- تمام حروف انگلیسی عنوان به حروف کوچک تبدیل می‌شوند.

۲- مجموعه n-تایی‌های موجود در عنوان به دست آمده و با توجه به معیار ژاکارد با مجموعه n-تایی‌های هر عنوان ذخیره شده در جدول اسناد مقایسه می‌شود. فرض کنید PM مجموعه شناسه‌های متناظر با اعضایی از جدول اسناد باشد که میزان تشابه مجموعه حروف متناظر با آن‌ها با مجموعه حروف عنوان مورد جستجو حداقل برابر با حداقل t_{tit}^{1} باشد. در آزمایشات محدود صورت گرفته، نتایج مناسبی با استفاده از n=3 و $t_{tit}^{1} = 0.6$ به دست آمده است. تعیین مقدار دقیق‌تر برای این پارامترها در گام بعدی صورت خواهد گرفت.

۳- به ازای هر شناسه موجود در PM:

a. با استفاده از الگوریتم لون‌اشتاین، فاصله بین مولفه عنوان متناظر با این شناسه و رشته X محاسبه می‌شود.

۴- مجموعه List که عبارت است از مجموعه‌ی شناسه‌هایی که میزان مشابهت عنوان متناظر با آن‌ها و X حداقل برابر با t_{tit}^{2} است، بازگردانده می‌شود. در آزمایشات صورت گرفته از ۰٫۵ به عنوان مقدار پارامتر t_{tit}^{2} استفاده شده و نتایج به دست آمده قابل قبول بوده است.

نمونه‌هایی از نتایج به دست آمده با استفاده از الگوریتم TitleMatch

در ادامه و در جدول ۱۳، نمونه‌هایی از نتایج حاصل از اعمال دسته‌بند TitileMatch روی تعدادی عنوان ورودی ارائه می‌شود. نمونه‌های موجود در این جدول، به صورت دستی و با شبیه‌سازی خطاهای ممکن در استخراج مولفه‌های یک رشته مرجع ایجاد شده‌اند.

جدول ١٤. نمونه‌هایی از تطابق‌های احتمالی تشخیص داده شده با استفاده از الگوریتم TitleMatch. دو ستون آخر این جدول، میزان شباهت به دست آمده با استفاده از معیارهای ژاکارد و لون‌اشتاین را گزارش می‌کنند.

تصمیم‌گیری چند معیاره	تصمیم‌گیری‌های چند معیاره
اصول هیدرولوژی کاربردی	اصول هیدرولوژی کاربردی
دیدگاه‌های نو در جغرافیایی شهری	دیدگاه‌های نو در جغرافیای شهری
الاشارات و التنبیهات	اشارات و تنبیهات
دانشنامه مدیریت شهری و روستایی	منظر شهری: در دانشنامه مدیریت شهری و روستایی
Business model innovation: Opportunities and Barriers	Business model innovation: Opportunities and Barriers
Numerical simulation of soil heat exchanger storage systems	Numerical simulation of soil heat exchanger storage systems for greenhouses
Cardiovascular effects pollution	Cardiovascular effects of air pollution

٤-٦-٤. تطبیق مراجع

همانطور که ذکر شد، به دلایل مختلفی من جمله قالب‌های مختلف استناددهی، اشتباهات کاربر، استفاده از اسامی مخفف در استنادها و ...، امکان بررسی تطبیقی رشته‌های مرجع به صورت ساده و با اعمال برای مثال یک الگوریتم تطبیق رشته تقریبی مثل فاصله لون‌اشتاین وجود ندارد. تحقیقات کمی در این باره انجام شده که در آن‌ها از برخی ملاک‌ها برای بررسی تطابق دو رشته مرجع استفاده شده است. در (Clegg and Pledge 2008) روشی بدین منظور ارائه شده که با توجه به مولفه‌های عنوان و نام نویسنده اول تطبیق دو رشته مرجع را بررسی می‌کند. در این پژوهش، از نسخه‌ای تغییریافته از این روش برای تطبیق رشته‌های مرجع استفاده می‌کنیم. تغییرات صورت گرفته در این روش بدین صورت هستند که در صورتی که با توجه به معیار ذکر شده در (Clegg and Pledge 2008) تطابق تشخیص داده نشد، از موارد اطلاعاتی دیگری مانند نام محل، سال انتشار، شماره صفحه و شماره دوره (برای مقالات چاپ شده در نشریه) استفاده می‌کنیم. با فرض در اختیار داشتن جدولی حاوی مولفه‌های استخراجی از پارساها و مراجع آن‌ها، الگوریتم CitMatch با ورودی یک رشته مرجع جدید به صورت زیر عمل می‌کند:

۱- زبان و نوع رشته مرجع جدید محاسبه می‌شود.

۲- با توجه به زبان رشته مرجع جدید، الگوریتم تجزیه رشته مرجع مناسب بر روی آن اعمال شده و اقلام اطلاعاتی آن استخراج می‌شود.

۳- به ازای هر سند ذخیره شده در جدول اسناد:

a. در صورت برابری زبان و نوع آن با زبان و نوع رشته مرجع جدید:

i. نویسنده اول آن با نویسنده اول رشته مرجع جدید مقایسه می‌شود. در صورت برابری، از الگوریتم TitleMatch برای مقایسه عناوین استفاده شده و در صورت تطابق، شناسه سند بازگردانده شده به یک لیست L اضافه می‌شود.

b. در صورت ناتهی بودن L، با توجه به عنوان و الگوریتم TitleMatch، شناسه شبیه‌ترین سند به عنوان تطابق بازگردانده می‌شود.

c. در غیر این صورت، به ازای هر سند موجود در جدول اسناد:

d. برابرای زبان و نوع سند با زبان و نوع رشته مرجع مقایسه می‌شود. در صورت برابری، سال انتشار و محل نشر آن‌ها با یکدیگر مقایسه می‌شود.

e. در ادامه، در صورتی که:

f. نویسنده‌ای مشترک بین مجموعه نویسندگان سند و رشته مرجع جدید وجود داشته باشد و میزان شباهت عناوین آن‌ها از مقداری بیشتر باشد، شناسه سند به عنوان تطابق بازگردانده می‌شود.

g. در صورتی که نوع رشته مرجع "مقاله نشریه" باشد و دوره و شماره صفحات آن با اطلاعات متناظر سند یکسان باشد، شماره سند به عنوان تطابق بازگردانده می‌شود.

h. null بازگردانده می‌شود.

ارزیابی الگوریتم پیشنهادی

برای ارزیابی روش پیشنهادی برای تطبیق رشته‌های مرجع، مجموعه داده‌ای شامل ۲۰۰ رشته مرجع جدید از پارساهای موجود در پایگاه گنج استخراج شده است. این مجموعه داده شامل ۱۰۰ رشته مرجع به زبان فارسی و ۱۰۰ رشته مرجع به زبان انگلیسی بوده که رشته‌های مرجع هر زبان نیز شامل ۴ دسته ۲۵ عددی از انواع مقاله نشریه، مقاله همایش، کتاب و پارسا است. در ادامه، با نگارش اسامی نویسندگان و محل‌ها به اشکال مختلف مرسوم، نوشتن سال

انتشار در موقعیت‌های متفاوت، حذف یا اضافه کردن برخی از اقلام اطلاعاتی مانند آدرس، شماره، شماره صفحه و ... در رشته مرجع و تغییرات در نقطه‌گذاری، هر یک از ۲۰۰ رشته مرجع موردنظر به ۳ شکل متفاوت نوشته شده است.

پس از آماده‌سازی مجموعه داده موردنظر، روش ارائه شده برای تطبیق رشته‌های مرجع بر روی آن‌ها اعمال شده است. نتایج حاصل به شرح زیر است:

جدول ۱۵. نتایج به دست آمده برای الگوریتم تطبیق رشته با توجه به مجموعه داده آزمایش ایجاد شده بدین منظور.

زبان	دقت	فراخوانی	F1
فارسی	۱	۰,۸۱	۰,۹۰
انگلیسی	۱	۰,۷۵	۰,۸۶
مجموع	۱	۰,۷۸	۰,۸۸

همانطور که مشخص است، دقت به دست آمده بسیار بالا است. یکی از دلایل دستیابی به این میزان دقت، کم بودن تعداد اسناد با اقلام اطلاعاتی مانند نویسنده یکسان است. در پیاده‌سازی روی مجموعه داده‌ای بزرگ، ممکن است مقدار به دست آمده برای دقت با افت محسوسی مواجه شود و لازم شود پارامترهای آستانه‌ای به کار رفته در این پژوهش را افزایش داد. لازم به ذکر است که با اعمال چنین تغییری، پارامتر فراخوانی کاهش خواهد یافت اما به نظر پژوهشگران در اینجا پارامتر دقت اهمیت بیشتری دارد.

٤-۷- افزودن پارساهای تهیه شده با فرمت tex. به شبکه استنادی

روش ارائه شده در بخش‌های قبل، برای افزودن پارساهای تهیه شده با نرم‌افزارهای word و open office ارائه شده است. استفاده از روش پیشنهادی برای افزودن پارساهای تهیه شده با فرمت tex. مقداری متفاوت است. تفاوت موجود در اغلب مواقع موجب تسهیل فرایند افزودن پارساهای تهیه شده با فرمت tex. به شبکه تحلیل استنادی می‌شود. تسهیل ایجاد شده به دلیل ساختارمند بودن فایل‌های tex است. برای درک بهتر این مساله، دو حالت استاندارد زیر برای ارجاع‌دهی در فایل‌های تک را در نظر بگیرید:

۱. قرار دادن مراجع در یک فایلbib.
۲. مشخص کردن یک به یک مراجع در فایل با استفاده از کلیدواژه bibitem\.

در حالت اول، استخراج مراجع، نوع مراجع و تجزیه مراجع به خودی خود و به طور کامل و دقیق وجود دارد و نیازی به انجام این مراحل نیست. در حالت دوم تنها استخراج مراجع بسیار ساده بوده و نیازی به استفاده از روش پیشنهادی در بخش‌های قبل برای این کار نیست. با توجه به این توضیحات، در ادامه، فرایند پیشنهادی برای اضافه کردن یک پارسای تهیه شده با فرمت tex. ارائه می‌شود. فرض کنیم فایل‌های متناظر با پارسای تهیه شده با فرمت tex. در قالب یک فایل فشرده[۱] باشند. با توجه به این فرض، برای افزودن این پارسا به شبکه تحلیل استنادی:

۱. فایل دریافتی را از حالت فشرده خارج می‌کنیم.
۲. در صورت وجود یک فایل با پسوند bib. در پوشه حاصل:

a) متن فایل با پسوند bib. را استخراج کرده و در یک رشته متنی قرار می‌دهیم.

i. در هر سطر موجود در متن (مشخص شده با کاراکتر Enter)، به دنبال کاراکتر "٪" گشته و در صورت وجود این کاراکتر، متن موجود پس از این کاراکتر را از رشته‌ی متنی حذف می‌کنیم.
ii. رشته‌ی متنی را با توجه به کاراکتر "@" به دنباله‌ای از رشته‌ها (که هر یک جزئیات یک رشته‌ی مرجع هستند) تقسیم‌بندی کرده و هر رشته با طول کمتر از ۱۰ را حذف می‌کنیم.
iii. به ازای هر رشته‌ی حاصل:

- نوع رشته مرجع با توجه به کلیدواژه استفاده شده بعد از کاراکتر "@" مشخص می‌شود.
- هر خط از رشته که شامل کاراکتر "=" باشد را با توجه به این کاراکتر به دو قسمت تقسیم نموده و با توجه به عبارت موجود در قسمت سمت چپ، برچسب قسمت سمت راست را مشخص می‌کنیم. برای درک بهتر این حالت، به مثال ۱ توجه شود.
- با توجه به الگوریتم‌های مربوطه ارائه شده در بخش‌های قبل، نوع هر رشته مرجع مشخص شده و تطبیق رشته‌های مرجع صورت می‌گیرد.

---

[۱] Zip

۳. فرض کنیم که پوشه حاصل شامل یک فایل با پسوند bib. نباشد. در این صورت:

a) رشته‌ی متنی متناظر با هر فایل با پسوند tex. موجود در پوشه را به دست می‌آوریم.

b) در هر سطر موجود در هر رشته‌ی متنی (مشخص شده با کاراکتر خط جدید (Enter))، به دنبال کاراکتر "٪" گشته و در صورت وجود این کاراکتر، متن موجود پس از این کاراکتر از رشته‌ی متنی حذف می‌شود.

c) مجموعه‌ای از اصلاحات به صورت زیر بر روی هر یک از رشته‌های مرجع حاصل اعمال می‌شود:

i. دستورهای "//"، "par\" و "item\" با دو کاراکتر "خط جدید" پیاپی جایگزین می‌شوند.

ii. قبل از هر دستور "bibitem\"، دو کاراکتر "خط جدید" پیاپی قرار داده می‌شود.

iii. در صورت وجود یک کاراکتر "خط جدید" بین دو سطر، آن کاراکتر حذف شده و دو سطر به یکدیگر متصل می‌شوند.

d) به ازای هر رشته‌ی متنی به دست آمده:

i. با توجه به کاراکتر "خط جدید"، هر سطر از رشته‌ی متنی که با کاراکتر "٪" آغاز شده باشد را حذف می‌کنیم.

ii. در هر رشته‌ی متنی، وجود دستور \begin{thebibliography} را بررسی می‌کنیم. در صورت وجود این دستور، تنها قسمت‌هایی از رشته‌ی متنی که بین دو عبارت \begin{thebibliography} و \end{thebibliography} قرار دارد را نگه داشته و مابقی بخش‌های آن را حذف می‌کنیم. در ادامه، با توجه به دستور bibitem\، متن حاصل به مجموعه‌ای از رشته‌های متنی متناظر با رشته‌های مرجع تقسیم شده و عملیات تجزیه رشته‌های مرجع، همانند آن‌چه برای یک رشته‌ی مرجع در بخش‌های قبل داشتیم انجام شده و سایر اقدامات موردنیاز صورت می‌گیرد. برای درک بهتر این حالت، به مثال ۲ توجه شود. لازم به ذکر است که در این حالت، برای پردازش هر رشته مرجع، عبارت }bibitem\ به علاوه ادامه متن تا زمان رسیدن به کاراکتر "{" از آن حذف می‌شوند.

iii. در صورت عدم وجود دستور \begin{thebibliography}، همانند آن‌چه برای استخراج رشته‌های مرجع از یک فایل ورد داشتیم، عمل کرده، استخراج رشته‌های مرجع و سایر مراحل انجام می‌شود.

مثال ۱: فرض کنیم فایل bib. موردنظر شامل دو رشته‌ی مرجع، به صورت زیر باشد:

```
@Article{Tassa2011,
author="Tassa, Tamir,"
title="Generalized oblivious transfer by secret sharing,"
journal="Designs, Codes and Cryptography,"
year="2011,"
volume="58,"
number="1,"
pages="11--21,"
}
@article{kn,
 title={Efficient k-out-of-n oblivious transfer scheme with the ideal communication cost},
 author={Lai, Jianchang and Mu, Yi and Guo, Fuchun and Chen, Rongmao and Ma, Sha},
 journal={Theoretical Computer Science},
 volume={714},
 pages={15—26},
 year={2018},
⁄publisher={Elsevier}
}
```

با توجه به فرایند ارائه شده، در ابتدا، با توجه به مرحله 2-a-i، خط

```
⁄publisher={Elsevie}
```

حذف می‌شود. در ادامه، با توجه به مرحله 2-a-ii، تفکیک رشته‌های مرجع انجام شده و دو رشته‌ی متنی زیر به دست می‌آید:

```
Article{Tassa2011,
author="Tassa, Tamir,"
title="Generalized oblivious transfer by secret sharing,"
```

journal="Designs, Codes and Cryptography,"

year="2011,"

volume="58,"

number="1,"

pages="11--21,"

}

و

article{kn,

title={Efficient k-out-of-n oblivious transfer scheme with the ideal communication cost},
author={Lai, Jianchang and Mu, Yi and Guo, Fuchun and Chen, Rongmao and Ma, Sha},
journal={Theoretical Computer Science},
volume={714},
pages={15—26},
year={2018},
}

سپس، با توجه به مرحله‌ی 2-a-iii، برای رشته‌ی متنی اول خواهیم داشت:

Type: article

Author: "Tassa, Tamir,"

Title: "Generalized oblivious transfer by secret sharing,"

Journal:"Designs, Codes and Cryptography,"

Year: "2011,"

Volume:"58,"

Number:"1,"

Pages: "11--21,"

به این ترتیب، اقلام اطلاعاتی متناظر با دیگر رشته‌های مرجع به دست می‌آید. پس از انجام این مراحل، با توجه به الگوریتم‌های مربوطه ارائه شده در بخش‌های قبل برای تعیین نوع رشته‌های مرجع و تطبیق رشته‌های مرجع استفاده می‌کنیم.

مثال ۲. فرض کنیم متن بین دو عبارت \begin{thebibliography} و \end{thebibliography} در یک فایل tex. به صورت زیر باشد:

\bibitem{conte} S D Conte, H E Dunsmore and V Y Shen, Software Engineering metrics and models, Benjamin/Cummings, 1986

\bibitem{troy} D A Troy and S H Zweben, Measuring the Quality of Structured Designs, Journal of Systems and Software, {\bf 2}, 1981, 113—120

در ادامه، دو رشته‌ی متنی زیر حاصل می‌شود:

S D Conte, H E Dunsmore and V Y Shen, Software Engineering metrics and models, Benjamin/Cummings, 1986

و

D A Troy and S H Zweben, Measuring the Quality of Structured Designs, Journal of Systems and Software, 2, 1981, 113—120.

سپس، هر یک از دو رشته فوق برای به دست آوردن اقلام اطلاعاتی مربوطه، به الگوریتم‌های ارائه شده در بخش‌های قبل ارسال می‌شوند.

نتیجه‌گیری و کارهای آینده

در این پژوهش، به مساله ساخت خودکار شبکه تحلیل استنادی بر روی مجموعه پارساهای پایگاه گنج پرداخته شد. در ابتدا پیشینه مساله و کارهای صورت گرفته برای تجزیه رشته‌های مرجع در زبان انگلیسی بررسی شد. در ادامه، از روش یادگیری ماشینی ماشین بردار پشتیبان و الگوریتم‌های تطبیق رشته تقریبی استفاده گشته و یک روش هوشمند جهت ساخت خودکار شبکه تحلیل استنادی با ورودی مجموعه‌ای از پارساهای دریافتی ارائه شد. روش پیشنهادی پیاده‌سازی شده و با استفاده از مجموعه داده‌های ایجاد شده در این پژوهش مورد آموزش و آزمایش قرار گرفت. نتایج آزمایشات صورت‌گرفته نشان‌دهنده کیفیت مناسب روش پیشنهادی است. پارامترهایی نهایی به دست آمده از ارزیابی روش پیشنهادی برابر با مقدار ۰,۹۹ برای پارامتر دقت، ۰,۷۳ برای پارامتر فراخوانی و ۰,۸۵ برای پارامتر F1 می‌باشد.

یکی از نقاط ضعف روش پیشنهادی این است که در آن، در اولین مشاهده رشته مرجع متناظر با یک سند، اقلام اطلاعاتی آن به دست آمده، در جدول اسناد ذخیره شده و دیگر تغییر نمی‌کند. با این حال، در بسیاری از موارد ارجاعات کامل نبوده یا تمام اقلام اطلاعاتی یک رشته مرجع به درستی توسط روش پیشنهادی شناسایی نمی‌شود و ممکن است در اقلام اطلاعاتی به دست آمده در مشاهدات بعدی از آن رشته مرجع کامل‌تر باشند. برای حل این مشکل می‌توان از مشاهدات بعدی یک رشته مرجع برای اصلاح جدول اسناد استفاده کرد. با این حال، با توجه به گسترده بودن بحث، بررسی این مساله را به پژوهش‌های آتی واگذار می‌کنیم.

همچنین، یکی از فرضیات در نظر گرفته شده در این پژوهش این است که فراداده‌های متناظر با پارساهای ورودی (شامل نام نویسنده، عنوان، دانشگاه یا پژوهشگاه محل انجام پژوهش و سال انجام پژوهش) همراه با آن‌ها به عنوان ورودی دریافت می‌شود. با توجه به اینکه در حال حاضر استخراج این فراداده‌ها توسط کاربر صورت می‌گیرد، یکی دیگر از کارهایی که در راستای بهبود این پژوهش می‌توان انجام داد، ارائه روشی است که به صورت خودکار این کار را انجام دهد.

از جمله دیگر کارهای قابل انجام برای ارتقا این پژوهش می‌توان به استفاده از دسته‌بندهای دیگر مانند ماشین بردار پشتیبان دوقلو[۱]، CRF، یادگیری عمیق و دسته‌بندهای ترکیبی برای حل مسائل در نظر گرفته شده در این پژوهش و مقایسه نتایج با نتایج به دست آمده در این پژوهش اشاره کرد.

[۱] Twin SVM

مراجع

نصیری، ج. (۱۳۹۰). بازشناسی اعمال انسان با رویکرد مقاوم سازی دسته‌بند تفکیکی. رساله دکتری دانشگاه تربیت مدرس، تهران.

کارگر، م. (۱۳۹۰). دسته‌بندی داده‌ها با استفاده از روش SVM. پایان‌نامه کارشناسی دانشگاه شهید باهنر کرمان.

پاک‌نیت، ن.، ج. نصیری، ع. جلالی‌منش (۱۳۹۷). طراحی یک روش هوشمند تجزیه رشته‌های مرجع در زبان فارسی. تهران: پژوهشگاه علوم و فناوری اطلاعات ایران.

فرهادی، م.، م. جم‌زاد (۱۳۹۷). بررسی معیارهای شباهت در بازیابی تصویر مبتنی بر محتوا. علوم رایانشی، ۱۳-۲۷.

Lawrence, S., C. Lee Giles, and K. Bollacker (1999). Digital libraries and autonomous citation indexing. Computer 32(6); 67-71,

Lawrence, S., C. L. Giles, and K. D. Bollacker (1999). Autonomous citation matching. In Proceedings of the third annual conference on Autonomous Agents; 392-393,

Huang, I. A., J. M. Ho, H. Y. Kao, and W. C. Lin (2004). Extracting citation metadata from online publication lists using BLAST. In Pacific-Asia Conference on Knowledge Discovery and Data Mining; 539-548,

Gupta, D., B. Morris, T. Catapano, and G. Sautter (2009). A new approach towards bibliographic reference identification, parsing and inline citation matching. In International Conference on Contemporary Computing; 93-102,

Chen, C. C., K. H. Yang, H. Y. Kao, and J. M. Ho (2008). BibPro: A Citation parser based on sequence alignment techniques. In Advanced Information Networking and Applications-Workshops, 22nd International Conference on; 1175-1180,

Seymore, K., A. McCallum, R. Rosenfeld, (1999). Learning hidden Markov model structure for information extraction. In AAAI-99 workshop on machine learning for information extraction; 37-42,

Besagni, D., A. Belaïd, N. Benet (2003). A segmentation method for bibliographic references by contextual tagging of fields. In Document Analysis and Recognition, 2003. Proceedings. Seventh International Conference on; 384-388,

Ding, Y., G. Chowdhury, and S. Foo (1999). Template mining for the extraction of citation from digital documents. In Proceedings of the Second Asian Digital Library Conference, Taiwan; 47-62,

Vinkler, P. (1994). The origin and features of information referenced in pharmaceutical patents. Scientometrics, 30(1); 283-302,

Constantin, A., S. Pettifer, and A. Voronkov (2013). PDFX: fully-automated PDF-to-XML conversion of scientific literature. In Proceedings of the 2013 ACM symposium on Document engineering; 177-180,

Hetzner, E. (2008). A simple method for citation metadata extraction using hidden markov models. In Proceedings of the 8th ACM/IEEE-CS joint conference on Digital libraries; 280-284,

Yin, P., M. Zhang, Z. Deng, and D. Yang (2004). Metadata extraction from bibliographies using bigram HMM. In International Conference on Asian Digital Libraries; 310-319,

Bikel, D. M., S. Miller, R. Schwartz, and R. Weischedel (1997). Nymble: a high-performance learning name-finder. In Proceedings of the fifth conference on applied natural language processing; 194-201,

Ojokoh, B., M. Zhang, and J. Tang (2011). A trigram hidden Markov model for metadata extraction from heterogeneous references. Information Sciences, 181(9); 1538-1551,

Lafferty, J., A. McCallum, and F. C. Pereira (2001). Conditional random fields: Probabilistic models for segmenting and labeling sequence data,

Peng, F., and A. McCallum (2013). Accurate information extraction from research papers using conditional random fields. Retrieved on April 13,

Councill, I. G., C. L. Giles, and M. Y. Kan (2008). ParsCit: an Open-source CRF Reference String Parsing Package. In LREC, 8; 661-667,

Zou, J., D. Le, and G. R. Thoma (2010). Locating and parsing bibliographic references in HTML medical articles. International Journal on Document Analysis and Recognition, 13(2); 107-119,

Zhang, Q., Y. G. Cao, and H. Yu (2011). Parsing citations in biomedical articles using conditional random fields. Computers in biology and medicine, 41(4); 190-194,

Kim, Y. M., P. Bellot, J. Tavernier, E. Faath, and M. Dacos (2012). Evaluation of BILBO reference parsing in digital humanities via a comparison of different tools. In Proceedings of the 2012 ACM symposium on Document engineering; 209-212,

Tkaczyk, D., P. Szostek, P. J. Dendek, M. Fedoryszak, and L. Bolikowski (2014). Cermine--automatic extraction of metadata and references from scientific literature. In Document Analysis Systems (DAS), 2014 11th IAPR International Workshop on; 217-221,

Tkaczyk, D., P. Szostek, M. Fedoryszak, P. J. Dendek, and L. Bolikowski, (2015). CERMINE: automatic extraction of structured metadata from scientific literature. International Journal on Document Analysis and Recognition (IJDAR), 18(4); 317-335,

Zhang, X., J. Zou, D. X. Le, and G. R. Thoma (2011). A structural SVM approach for reference parsing. BMC bioinformatics, 12(3); S7,

Agichtein, E., and V. Ganti (2004). Mining reference tables for automatic text segmentation. In Proceedings of the tenth ACM SIGKDD international conference on Knowledge discovery and data mining; 20-29,

Lopez, P. (2009). GROBID: Combining automatic bibliographic data recognition and term extraction for scholarship publications. In International Conference on Theory and Practice of Digital Libraries; 473-474,

Settles, B. (2004). Biomedical named entity recognition using conditional random fields and rich feature sets. In Proceedings of the international joint workshop on natural language processing in biomedicine and its applications; 104-107,

Vapnik, V. N. (1998). Statistical Learning Theory. John Wiley & Sons, New York,

Vapnik, V. N. (1995). The Nature of Statistical Learning Theory. Springer-Verlag, New York,

Liu, B. (1998). Minimax chance constrained programming models for fuzzy decision system. Information Science, 112; 25-38,

Masulli, F., and G. Valentini (2000). Comparing decomposition methods for classification. Proceedings of the Fourth International Conference on Knowledge- Based Intelligent Engineering Systems and Allied Technologies (KES 2000), Brighton, UK, volume 2; 788–791,

Krebel, U. H. G. (1999). Pairwise classification and support vector machines. In B. Scholkopf, C. J. C. Burges, and A. J. Smola, editors, Advances in Kernel Methods: Support Vector Learning; 255–68,

Amari, S., S. Wu (1999). An information-geometrical method for improving the performance of support vector machine classifiers. In Proceedings of the Ninth International Conference on Artificial Neural Networks (ICANN ’99), Edinburgh, UK, volume 1; 85–90,

LeCun, Y., L. Bottou, Y. Bengio, and P. Haffner (1998). Gradient-based learning applied to document recognition. Proceedings of the IEEE, 86(11); 2278–324,

Bradley, P. S., and O. L. Mangasarian (1998). Feature selection via concave minimization and support vector machines. In Proceedings of the Fifteenth International Conference on Machine Learning (ICML ’98), Madison; 82–90,

Wang, P., B. Wu, X. Li, L. Wang, and B. Wang (2015). A Simhash-Based Generalized Framework for Citation Matching in MapReduce. In Trends and Applications in Knowledge Discovery and Data Mining; 78-90,

Prasad, A. S., and S. Rao (2010). Citation matching in Sanskrit corpora using local alignment. In International Sanskrit Computational Linguistics Symposium; 124-136,

Pasula, H., B. Marthi, B. Milch, S. J. Russell, and I. Shpitser (2003). Identity uncertainty and citation matching. In Advances in neural information processing systems; 1425-1432,

Clegg, A. B., and D. Pledge (2008). Streamlining the Clinical Guideline Production Process with Fuzzy Citation Matching. In The Proceedings of the First Conference on Text and Data Mining of Clinical Documents (Louhi'08); 3-22,

پیوست ۱: کد پایتون روش ارائه شده

در این قسمت، کدهای نوشته شده در پیاده‌سازی آزمایشی روش پیشنهادی آورده شده است. این پیاده‌سازی با استفاده از زبان برنامه‌نویسی پایتون صورت گرفته و کدهای ارائه شده در این بخش این کدها هستند. لازم به ذکر است که پس از پیاده‌سازی آزمایشی روش پیشنهادی، پیاده‌سازی نهایی آن توسط متخصص برنامه‌نویسی و به زبان c# صورت گرفته اما در اینجا از ارائه این کد صرف‌نظر شده است:

```
import time
import docx2txt
import logging
import matplotlib.pyplot as plt
import os
import codecs
import numpy as np
from unidecode import unidecode
from random import randint
import pandas as pd
from sklearn.svm import SVC
from sklearn.metrics import classification_report, confusion_matrix
import math
from xlrd import *
from openpyxl import load_workbook
current_dir=os.getcwd()
files_dir=current_dir+"\Data"
common_words_in_venues=["دانشگاه","پژوهشگاه","استان","شهید","university","school","انتشارات","نشر","ناشر","الانتشار","چاپ","جهاد",
                        "pub","inc","press","publishing","publication","publications","فصلنامه","پژوهشنامه","مجله","دوفصلنامه",
                        "نشریه","دوماهنامه","ماهنامه","پژوهش‌نامه","فصل‌نامه","دوفصل‌نامه","مانامه","پژوهش","تحقیقات","تازه‌هاي",
                        "مطالعات","journal","j","research","international","int","letters","let","science","sciences","sci","review",
                        "advances","bulletin","همایش","کنفرانس","ایران","ملی","بین‌المللی","کنگره","سمینار","انجمن","conference",
                        "symposium","congress","proceedings","annual","seminar"]
numbers=["0","1","2","3","4","5","6","7","8","9"]

seperators_Both=["،",":",",",",",".","؛","(",")","\"","»","«","؟","?","(",")",";","[","]",":","&"]
seperators_Per=["،",".",":",":","؛","(",")","\"","»","«","؟","?"]
seperators_Eng=["،",",",",",".","؛","(",")","\"","»","«","؟","?","(",")",";","[","]"]
seperators=["،",":",",",",","؛",";","&","-"]
LL=["0","J","C","B","T"]
seyed=["سید","سیده","السادات"]

citations_table=[]
documents_table=[]
authors_table=[]
venues_table=[]

engalph=["a","b","c","d","e","f","g","h","i","j","k","l","m","n","o",
         "p","q","r","s","t","u","v","w","x","y","z"]
ENGALPH=["A","B","C","D",
         "E","F","G","H","I","J","K","L","M","N","O","P","Q","R","S",
         "T","U","V","W","X","Y","Z"]
peralph=["ا", "آ", "ب", "پ", "ت", "ث", "ج", "چ", "ح",
          "خ", "د", "ذ", "ر", "ز", "ژ", "س", "ش", "ص",
          "ض", "ط", "ظ", "ع", "غ", "ف", "ق", "ک", "گ",
          "ل", "م", "ن", "و", "ه", "ی","ي",""]
Fnameindicators_Eng=["a.","b.","c.","d.","e.","f.","g.","h.","i.","j.","k.","l.","m.",
                     "n.","o.","p.","q.","r.","s.","t.","u.","v.","w.","x.","y.","z."]
Fnameindicators_Per=["ا.", "آ.", "ب.", "پ.", "ت.", "ث.", "ج.", "چ.", "ح.",
                     "خ.", "د.", "ذ.", "ر.", "ز.", "ژ.", "س.", "ش.", "ص.",
                     "ض.", "ط.", "ظ.", "ع.", "غ.", "ف.", "ق.", "ک.", "گ.",
                     "ل.", "م.", "ن.", "و.", "ه.", "ی."]
lnameindicators=["پور","فرد","نیا","زاده","نژاد","راد","فر","ان","کیا","پناه","منش"]


svclassifier_ext_refs = SVC(C=4,kernel='linear')
svclassifier_Type_Eng = SVC(C=4,kernel='linear')
svclassifier_Type_Per = SVC(C=4,kernel='linear')
svclassifier_Parse_Eng = SVC(C=4,kernel='linear')
```

```
svclassifier_Parse_Per = SVC(C=4,kernel='linear')
svclassifier_NameLabeling = SVC(C=4,kernel='linear')####NNNN####

class Document:
    def __init__(self, doc_lang, doc_type, doc_authors,doc_title,doc_year,doc_venue,doc_volume,doc_pages,
doc_address,
                 doc_seri,doc_month,doc_text):
        self.lang=doc_lang
        self.type=doc_type
        self.authors = doc_authors
        self.title=doc_title
        self.year=doc_year
        self.venue=doc_venue
        self.volume=doc_volume
        self.pages=doc_pages
        self.address=doc_address
        self.seri=doc_seri
        self.month=doc_month
        self.text=doc_text

def FinSepAuths(seped_a):
    seped_auths=[]
    for s_a in seped_a:
        if not isinstance(s_a,list):
            tempstr=""
            list_chars=list(s_a)
            tempstr=tempstr+list_chars[0].upper()
            for i in list(range(1,len(list_chars))):
                if list_chars[i-1] in [".","-",",","،","؛",";","@"," "]:
                    tempstr=tempstr+list_chars[i].upper()
                else:
                    tempstr=tempstr+list_chars[i]
            for ch in [".","-",",","،","؛",";"]:
                tempstr=tempstr.replace(ch," ")
            tempstr=tempstr.replace(" @","@")
            tempstr=tempstr.replace("  "," ")
            while tempstr[-1:]==" ":
                tempstr=tempstr[:-1]
            seped_auths.append(tempstr)
        else:
            temp=FinSepAuths(s_a)
            for temp_auth in temp:
                seped_auths.append(temp_auth)
    return seped_auths
def sep_auths_assign_lbls(Author):
    auths_feild_features=Extract_features_NameLabeling(Author)
    auths_lbls=[]
    auths_tokens=[]
    for i in list(range(len(auths_feild_features))):
        new_row=[]
        if auths_feild_features[i][1][0]==1:
            temp=["0","0"]
        else:
            temp=[temp[1],last_lbl]
        for j in list(range(2)):
            new_row.append(temp[j])
        for j in list(range(len(auths_feild_features[0][1]))):
            new_row.append(auths_feild_features[i][1][j])
        if i<len(auths_feild_features)-1:
            for j in list(range(1, len(auths_feild_features[0][1]))):
                new_row.append(auths_feild_features[i+1][1][j])
```

```
        else:
            for j in list(range(1,len(auths_feild_features[0][1]))):
                new_row.append("0")
        df2=pd.DataFrame([new_row])
        x2=df2.drop(df2.columns[[]],axis=1)
        y_pred = svclassifier_NameLabeling.predict(x2)
        auths_tokens.append(auths_feild_features[i][0])
        last_lbl=y_pred[0]
        auths_lbls.append(last_lbl)
    return [auths_tokens,auths_lbls]
def sep_auths(tokens_lbls):
    auths_tokens=tokens_lbls[0]
    auths_lbls=tokens_lbls[1]
    seped_auths=[]
    auths=[]
    lbls=[]
    for i in list(range(len(auths_lbls))):
        if auths_lbls[i]!=0:
            auths.append(auths_tokens[i])
            lbls.append(auths_lbls[i])
        else:
            seped_auths.append(sep_auths([auths,lbls]))
            auths=[]
            lbls=[]
    if len(lbls)==0:
        x=""
    elif len(lbls)==1:
        seped_auths.append(auths[0])
    elif len(lbls)==2:
        if lbls[0]==2:
            seped_auths.append(str(auths[0])+"@"+str(auths[1]))
        else:
            seped_auths.append(str(auths[1])+"@"+str(auths[0]))
    elif len(lbls)==3:
        if lbls[0]==lbls[1]:
            if lbls[0]==2:
                seped_auths.append(str(auths[0])+" "+str(auths[1])+"@"+str(auths[2]))
            else:
                seped_auths.append(str(auths[2])+"@"+str(auths[0])+" "+str(auths[1]))
        elif lbls[1]==lbls[2]:
            if lbls[0]==2:
                seped_auths.append(str(auths[0])+"@"+str(auths[1])+" "+str(auths[2]))
            else:
                seped_auths.append(str(auths[1])+" "+str(auths[2])+"@"+str(auths[0]))
        else:
            auths1=[auths[0],auths[1]]
            lbls1=[lbls[0],lbls[1]]
            auths2=[auths[2]]
            lbls2=[lbls[2]]
            seped_auths.append(sep_auths([auths1,lbls1]))
            seped_auths.append(sep_auths([auths2,lbls2]))
    elif lbls==[2,1,2,1] or lbls==[1,2,1,2]:
        auths1=[auths[0],auths[1]]
        lbls1=[lbls[0],lbls[1]]
        auths2=[auths[2],auths[3]]
        lbls2=[lbls[2],lbls[3]]
        seped_auths.append(sep_auths([auths1,lbls1]))
        seped_auths.append(sep_auths([auths2,lbls2]))
    elif lbls==[2,2,1,1] or lbls==[1,1,2,2]:
        auths1=[str(auths[0])+" "+str(auths[1]),str(auths[2])+" "+str(auths[3])]
        lbls1=[lbls[0],lbls[2]]
```

```
        seped_auths.append(sep_auths([auths1,lbls1]))
    elif lbls==[2,2,2,1] or lbls==[1,1,1,2]:
        auths1=[str(auths[0])+" "+str(auths[1])+" "+str(auths[2]),str(auths[3])]
        lbls1=[lbls[0],lbls[3]]
        seped_auths.append(sep_auths([auths1,lbls1]))
    elif lbls==[2,1,1,1] or lbls==[1,2,2,2]:
        auths1=[str(auths[0]),str(auths[1])+" "+str(auths[2])+" "+str(auths[3])]
        lbls1=[lbls[0],lbls[3]]
        seped_auths.append(sep_auths([auths1,lbls1]))
    elif lbls==[2,1,1,2]:
        auths1=[auths[0],auths[1]]
        lbls1=[lbls[0],lbls[1]]
        auths2=[auths[3],auths[2]]
        lbls2=[lbls[3],lbls[2]]
        seped_auths.append(sep_auths([auths1,lbls1]))
        seped_auths.append(sep_auths([auths2,lbls2]))
    else:
        auths1=[]
        lbls1=[]
        auths2=[]
        lbls2=[]
        chng=0
        auths1.append(auths[0])
        lbls1.append(lbls[0])
        for k in list(range(1,len(lbls))):
            if lbls[k]!=lbls1[len(lbls1)-1]:
                chng+=1
            if chng<2 and len(lbls1)<3:
                auths1.append(auths[k])
                lbls1.append(lbls[k])
            else:
                auths2.append(auths[k])
                lbls2.append(lbls[k])
        seped_auths.append(sep_auths([auths1,lbls1]))
        seped_auths.append(sep_auths([auths2,lbls2]))
    fin_seped_auths=FinSepAuths(seped_auths)
    return fin_seped_auths

def xlsx_to_list(wb):
    tempp=[]
    sheet=wb.active
    max_row=sheet.max_row
    max_column=sheet.max_column
    for i in range(1,max_row+1):
        temparr=[]
        for j in range(1,max_column+1):
            temparr.append(sheet.cell(row=i,column=j).value)
        tempp.append(temparr)
    return tempp
def is_f_name_indicator(token,Fnameindicators):
    if token in Fnameindicators:
        return True
    else:
        return False
def is_contain_nonalphabetic(token,alph):
    for c in token:
        if c not in alph:
            return True
    return False

def is_contain_num_alph(token,alph):
```

```
    flag1=False
    flag2=False
    for c in token:
        if c in alph:
            flag1=True
    for c in token:
        if c in numbers:
            flag2=True
    return flag1 and flag2
def is_only_contain_num_alph(token,alph):
    for c in token:
        if not c in alph+numbers:
            return False
    return True
def is_et_al_Per(token):
    etalforms=["همکاران","دیگران"]
    if token in etalforms:
        return True
    else:
        return False
def is_j_ind_Per(token):
    inds=["فصلنامه","پژوهشنامه","مجله","دوفصلنامه","نشریه","دوماهنامه","ماهنامه","پژوهش‌نامه","فصل‌نامه","دوفصل‌نامه","ماهنامه","پژوهش",
"تحقیقات","تازه‌هاي","مطالعات"]
    tmp=token
    if token[-2:]=="ي":
        tmp=tmp[:-2]
    for ind in inds:
        if (tmp in inds) or (tmp.startswith(ind)):
            return True
    return False
def is_c_ind_Per(token):
    inds=["همایش","کنفرانس","ایران","ملي","بین‌المللي","کنگره","سمینار","انجمن","اولین","نخستین","یکمین","دومین","سومین","چهارمین","
پنجمین","ششمین","هفتمین","هشتمین","نهمین","دهمین","یازدهمین","دوازدهمین","سیزدهمین","چهاردهمین","پانزدهمین","شانزدهمین",
        "هفدهمین","هجدهمین","نوزدهمین","خلاصه","مقالات"]
    tmp=""
    token2=token
    if token2[-2:]=="ي":
        token2=token2[:-2]
    if "" in token2:
        parts=token2.split("")
        t1=parts[0]
        t2=parts[1]
        if len(t1)<len(t2):
            tmp=t2
        else:
            tmp=t1
    else:
        tmp=token2
    if tmp in inds:
        return True
    else:
        return False
def is_author_name_Per(token,names):
    if token in names:
        return True
    return False
def j_name_score_Per(token,jnames_scores):
    for js in jnames_scores:
        jss=js.split(": ")
        if jss[0]==token and len(jss)>1:
            return math.ceil(math.log(int(jss[1])))
```

```
    return 0
def c_name_score_Per(token,cnames_scores):
    for cs in cnames_scores:
        css=cs.split(": ")
        if css[0]==token and len(css)>1:
            return math.ceil(math.log(int(css[1])))
    return 0
def title_name_score_Per(token,tw_scores):
    n1=0
    n2=0
    n3=0
    n4=0
    if "" in token:
        partss=token.split("")
    for w in tw_scores:
        ws=w.split(": ")
        if len(ws)==2 and ws[0]==token:
            n3=int(ws[1])
        if "" in token:
            if len(ws)==2 and ws[0]==partss[0]:
                n1=int(ws[1])
            if len(ws)==2 and ws[0]==partss[1]:
                n2=int(ws[1])
            n4=min(n1,n2)
    return math.ceil(math.log(max(n4,n3)+1))
def is_in_pn_Eng(token,pnames):
    if token in pnames:
        return True
    return False
def is_in_jn_Eng(token,jnames):
    if token in jnames:
        return True
    return False
def is_in_cn_Eng(token,cnames):
    if token in cnames:
        return True
    return False

def is_containing_Eng_alphabet(token):
    for c in token:
        if c in engalph+ENGALPH:
            return True
    return False
def find_ind(l_p):
    ind=0
    maxx=0
    for i in list(range(30,len(l_p))):
        num=0
        for j in list(range(30)):
            if l_p[i-j]==1:
                num+=1
        if num>maxx:
            maxx=num
            ind=i-15
    return ind

def edit_array(Lbls,mean_ind):
    ind1=0
    for i in list(range(4,mean_ind)):
        flag=True
        for j in list(range(5)):
```

```
            if Lbls[i-j]==0:
                flag=False
        if flag:
            ind1=i-4
            break

    ind2=mean_ind
    for i in list(range(len(Lbls)-3,mean_ind,-1)):
        flag=True
        for j in list(range(3)):
            if Lbls[i+j]==0:
                flag=False
        if flag:
            ind2=i+2
            break
    Labels=[]
    for i in list(range(len(Lbls))):
        if ind1<=i<=ind2:
            Labels.append(Lbls[i])
        else:
            Labels.append(0)
    for i in list(range(ind1+4,ind2-4)):
        num_1_bef=0
        num_1_aft=0
        for j in list(range(4,0,-1)):
            if (Labels[i-j]==1):
                num_1_bef+=1
        for j in list(range(4,0,-1)):
            if (Labels[i+j]==1):
                num_1_aft+=1
        if (num_1_bef>=2 and num_1_aft>=2):
            Labels[i]=1
    for i in list(range(ind1+4,ind2-4)):
        num_1_bef=0
        num_1_aft=0
        for j in list(range(4,0,-1)):
            if (Labels[i-j]==1):
                num_1_bef+=1
        for j in list(range(4,0,-1)):
            if (Labels[i+j]==1):
                num_1_aft+=1
        if (num_1_bef>=2 and num_1_aft>=2):
            Labels[i]=1
    return Labels
def Virayesh(text):
    text=convertnumbs(text)
    text=text.replace("  "," ")
    text=text.replace("  "," ")
    text=text.replace("”"," ")
    text=text.replace("”"," ")
    text=text.replace("  "," ")
    text=text.replace("  "," ")
    text=text.replace("et al","etal")
    text=text.replace("–","-")
    text=text.replace("-","-")
    text=text.replace("  "," ")
    text=text.replace("“"," ")
    text=text.replace("”"," ")
    text=text.replace("&"," & ")
    text=text.replace("ﻁ","ﻃ")
    text=text.replace("ﺑ","ﺏ")
```

```
text=text.replace("ٱ","ا")
text=text.replace("ۑ","ي")
text=text.replace("ى","ي")
text=text.replace("ئ","ي")
text=text.replace("ی","ي")
text=text.replace("ي","ي")
text=text.replace("ى","ي")
text=text.replace("ئ","ي")
text=text.replace("ئ","ي")
text=text.replace("ي","ي")
text=text.replace("ي","ي")
text=text.replace("آ","ا")
text=text.replace("أ","ا")
text=text.replace("إ","ا")
text=text.replace("آ","ا")
text=text.replace("ا","ا")
text=text.replace("آ","ا")
text=text.replace("أ","ا")
text=text.replace("إ","ا")
text=text.replace("ا","ا")
text=text.replace("ب","ب")
text=text.replace("ب","ب")
text=text.replace("ﺑ","ب")
text=text.replace("ب","ب")
text=text.replace("پ","پ")
text=text.replace("ة","ه")
text=text.replace("ة","ه")
text=text.replace("ه","ه")
text=text.replace("ة","ه")
text=text.replace("ه","ه")
text=text.replace("ة","ه")
text=text.replace("ة","ه")
text=text.replace("ه","ه")
text=text.replace("ھ","ه")
text=text.replace("ت","ت")
text=text.replace("ﺗ","ت")
text=text.replace("ﺘ","ت")
text=text.replace("ﺖ","ت")
text=text.replace("ث","ث")
text=text.replace("ج","ج")
text=text.replace("چ","چ")
text=text.replace("ح","ح")
text=text.replace("ﺤ","ح")
text=text.replace("خ","خ")
text=text.replace("د","د")
text=text.replace("ﺪ","د")
text=text.replace("ﺪ","د")
text=text.replace("ذ","ذ")
text=text.replace("ﺪ","د")
text=text.replace("ﺬ","ذ")
text=text.replace("ر","ر")
text=text.replace("ۑ","ي")
text=text.replace("ﻣ","م")
text=text.replace("ﺣ","ح")
text=text.replace("ﻣ","م")
text=text.replace("د","د")
text=text.replace("ضـ","ض")
text=text.replace("ﻋ","ع")
text=text.replace("د","د")
text=text.replace("ر","ر")
text=text.replace("ﺳ","س")
```

```
    text=text.replace("و","و")
    text=text.replace("ﻟ","ل")
    text=text.replace("ﻛ","ک")
    text=text.replace("ن","ن")
    text=text.replace("ﺣ","ح")
    text=text.replace("غ","غ")
    text=text.replace("ﻓ","ف")
    text=text.replace("ﻋ","ع")
    text=text.replace("ﻓ","ف")
    text=text.replace("ت","ت")
    text=text.replace("ﻋ","ع")
    text=text.replace("ﻳ","ي")
    text=text.replace("ﺟ","ج")
    text=text.replace("ي","ي")
    text=text.replace("ى","ي")
    text=text.replace("١","1")
    text=text.replace("ء","")
    text=text.replace("٢","2")
    text=text.replace("٣","3")
    text=text.replace("٤","4")
    text=text.replace("٥","5")
    text=text.replace("٨","8")
    text=text.replace("٧","7")
    text=text.replace("٦","6")
    text=text.replace("ي","ي")
    text=text.replace("و","و")
    text=text.replace("\ufeff","")
    text=text.replace("ي","ي")
    text=text.replace("5","5")
    text=text.replace("¬","-")
    text=text.replace(" ،","،")
    text=text.replace(" .",".")
    text=text.replace("","")
    text=text.replace("  "," ")
    text=text.replace("ﺗ","ت")
    text=text.replace("-","")
    text=text.replace("ﻪ","ه")
    text=text.replace("ﺗ","ت")
    text=text.replace("ﮔ","گ")
    text=text.replace("ﻟ","ل")
    text=text.replace("ﺷ","ش")
    text=text.replace("ﻫ","ه")
    text=text.replace("ﻧ","ن")
    text=text.replace("ﺧ","خ")
    text=text.replace("ط","ط")
    text=text.replace("م","م")
    text=text.replace("ي","ي")
    text=text.replace("س","س")
    text=text.replace("-","-")
    text=text.replace(" هاي "," هاي")
    text=text.replace(" ها "," ها")
    text=text.replace(" ي "," ي")
    text=text.replace(" اي "," اي")
    text=text.replace("–","-")
    text=text.replace("ﻻ","لا")
    text=text.replace(" -","-")
    text=text.replace("- ","-")
    text=text.replace(" "," ")
    return text
def replace_sep_with_sepspace(text,seperators):
    tmp=text
```

```
    for s in seperators:
        tmp=tmp.replace(str(s), str(s)+" ")
    return tmp

def Virayesh_Both(text):
    text=special_year_editting(text)
    for ch in ENGALPH:
        text=text.replace(ch, ch.lower())
    text=convertnumbs(text)
    for c in engalph:
        text=text.replace(c+"-",c+" ")
        text=text.replace("-"+c," "+c)
    for n in numbers:
        text=text.replace(str(n)+" th ",str(n)+"th ")
        text=text.replace(str(n)+" -",str(n)+"-")
        text=text.replace("- "+str(n),"-"+str(n))
    text=text.replace(str(2)+" nd ",str(2)+"nd ")
    text=text.replace(str(1)+" st ",str(1)+"st ")
    text=text.replace(" "," ")
    text=text.replace("procedings of","proceedingsof")
    text=text.replace("proceding of","procedingof")
    text=text.replace("proceedings of","proceedingsof")
    text=text.replace("proceeding of","proceedingof")
    text=text.replace("new york","newyork")
    text=text.replace(" "," ")
    text=text.replace("”"," ")
    text=text.replace("”"," ")
    text=text.replace(" "," ")
    text=text.replace(" "," ")
    text=text.replace("et al","etal")
    text=text.replace("–","-")
    text=text.replace("-","-")
    text=text.replace(" "," ")
    text=text.replace("“"," ")
    text=text.replace("’"," ")
    text=text.replace("&"," & ")
    text=text.replace("m.sc","msc")
    text=text.replace("ph.d","phd")
    text=text.replace("ph. d.","phd")
    text=sep_text_from_number(text)
    for sep in seperators_Both:
        text=text.replace(sep,sep+" ")
    for sep in seperators_Both:
        while " "+sep in text:
            text=text.replace(" "+sep,sep)
    text=text.replace(":",": ")
    while " :" in text:
        text=text.replace(" :",":")
    while "  " in text:
        text=text.replace("  "," ")
    while "  " in text:
        text=text.replace("  "," ")
    for i1 in numbers:
        for i2 in numbers:
            text=text.replace(str(i1)+" -"+str(i2),str(i1)+"-"+str(i2))
            text=text.replace(str(i1)+"- "+str(i2),str(i1)+"-"+str(i2))
    text=text.replace("ﻁ","ط")
    text=text.replace("ﺒ","ب")
    text=text.replace("ﺎ","ا")
    text=text.replace("ﻲ","ي")
    text=text.replace("ى","ي")
```

```
606.  text=text.replace("ئ","ي")
607.  text=text.replace("ى","ي")
608.  text=text.replace("ي","ي")
609.  text=text.replace("ى","ي")
610.  text=text.replace("ئ","ي")
611.  text=text.replace("ئ","ي")
612.  text=text.replace("ي","ي")
613.  text=text.replace("ي","ي")
614.  text=text.replace("آ","ا")
615.  text=text.replace("أ","ا")
616.  text=text.replace("إ","ا")
617.  text=text.replace("أ","ا")
618.  text=text.replace("ا","ا")
619.  text=text.replace("آ","ا")
620.  text=text.replace("أ","ا")
621.  text=text.replace("إ","ا")
622.  text=text.replace("ا","ا")
623.  text=text.replace("ب","ب")
624.  text=text.replace("ب","ب")
625.  text=text.replace("پ","ب")
626.  text=text.replace("ب","ب")
627.  text=text.replace("پ","پ")
628.  text=text.replace("ة","ه")
629.  text=text.replace("ة","ه")
630.  text=text.replace("ه","ه")
631.  text=text.replace("ة","ه")
632.  text=text.replace("ه","ه")
633.  text=text.replace("ة","ه")
634.  text=text.replace("ة","ه")
635.  text=text.replace("ه","ه")
636.  text=text.replace("ۀ","ه")
637.  text=text.replace("ت","ت")
638.  text=text.replace("ٹ","ت")
639.  text=text.replace("ټ","ت")
640.  text=text.replace("ټ","ت")
641.  text=text.replace("ٹ","ٹ")
642.  text=text.replace("ج","ج")
643.  text=text.replace("چ","چ")
644.  text=text.replace("ح","ح")
645.  text=text.replace("ځ","ح")
646.  text=text.replace("خ","خ")
647.  text=text.replace("د","د")
648.  text=text.replace("ډ","د")
649.  text=text.replace("ډ","د")
650.  text=text.replace("ذ","د")
651.  text=text.replace("ډ","د")
652.  text=text.replace("ذ","ذ")
653.  text=text.replace("ر","ر")
654.  text=text.replace("ړ","ي")
655.  text=text.replace("م","م")
656.  text=text.replace("ح","ح")
657.  text=text.replace("م","م")
658.  text=text.replace("د","د")
659.  text=text.replace("ض","ض")
660.  text=text.replace("ع","ع")
661.  text=text.replace("د","د")
662.  text=text.replace("ر","ر")
663.  text=text.replace("س","س")
664.  text=text.replace("و","و")
665.  text=text.replace("ل","ل")
666.  text=text.replace("ک","ک")
```

```
667.  text=text.replace("ن","ن")
668.  text=text.replace("ح","ح")
669.  text=text.replace("غ","غ")
670.  text=text.replace("ﻓ","ف")
671.  text=text.replace("ﻋ","ع")
672.  text=text.replace("ﻓ","ف")
673.  text=text.replace("ت","ت")
674.  text=text.replace("ﻋ","ع")
675.  text=text.replace("ﻳ","ي")
676.  text=text.replace("ﺟ","ج")
677.  text=text.replace("ي","ي")
678.  text=text.replace("ى","ي")
679.  text=text.replace("١","1")
680.  text=text.replace("ء","")
681.  text=text.replace("٢","2")
682.  text=text.replace("٣","3")
683.  text=text.replace("٤","4")
684.  text=text.replace("٥","5")
685.  text=text.replace("٨","8")
686.  text=text.replace("٧","7")
687.  text=text.replace("٦","6")
688.  text=text.replace("ي","ي")
689.  text=text.replace("و","و")
690.  text=text.replace("\ufeff","")
691.  text=text.replace("ي","ي")
692.  text=text.replace("5","5")
693.  text=text.replace("¬","-")
694.  text=text.replace(" ،","،")
695.  text=text.replace(" .",".")
696.  text=text.replace("","")
697.  text=text.replace("  "," ")
698.  text=text.replace("ﺗ","ت")
699.  text=text.replace("-","")
700.  text=text.replace("ﻪ","ه")
701.  text=text.replace("ﺗ","ت")
702.  text=text.replace("ﮔ","گ")
703.  text=text.replace("ﻟ","ل")
704.  text=text.replace("ﺷ","ش")
705.  text=text.replace("ﻫ","ه")
706.  text=text.replace("ﻧ","ن")
707.  text=text.replace("ﺧ","خ")
708.  text=text.replace("ﻃ","ط")
709.  text=text.replace("ﻣ","م")
710.  text=text.replace("ي","ي")
711.  text=text.replace("ﺳ","س")
712.  text=text.replace("ـ","-")
713.  text=text.replace(" هاي "," هاي")
714.  text=text.replace(" ها "," ها")
715.  text=text.replace(" ي "," ي")
716.  text=text.replace(" اي "," اي")
717.  text=text.replace("–","-")
718.  text=text.replace("ﻻ","لا")
719.  text=text.replace(" -","-")
720.  text=text.replace("- ","-")
721.  text=text.replace("بامقدمه","با مقدمه")
722.  text=text.replace("دانشكده‌ي","دانشکده")
723.  text=text.replace("ترجمهي","ترجمه")
724.  text=text.replace("به‌كوشش","به کوشش")
725.  text=text.replace("پژوهش‌نامه","پژوهش نامه")
726.  text=text.replace("فصل‌نامه","فصل نامه")
727.  text=text.replace("فصلنامه","فصل‌نامه")
```

```
    text=text.replace("بين المللي","بين‌المللي")
    text=text.replace("پايان نامه","پايان‌نامه")
    text=text.replace("پايانناامه","پايان‌نامه")
    text=text.replace("پايان‌نامه‌ي","پايان‌نامه")
    text=text.replace("رسالهٔ","رساله")
    text=text.replace("رساله‌ي","رساله")
    text=text.replace("رسالهي","رساله")
    text=text.replace("مجلهي","مجله")
    text=text.replace("مجله‌ي","مجله")
    text=text.replace("دوره‌ي","دوره")
    text=text.replace("شماره‌ي","شماره")
    text=text.replace("دورهي","دوره")
    text=text.replace("شمارهي","شماره")
    text=text.replace("-","")
    text=dash_editor(text)
    return text

def Virayesh_Per(text):
    text=special_year_editting(text)
    text=convertnumbs(text)
    for n in numbers:
        text=text.replace(str(n)+" -",str(n)+"-")
        text=text.replace("- "+str(n),"-"+str(n))
    text=sep_text_from_number(text)
    for sep in seperators_Per:
        text=text.replace(sep,sep+" ")
    for sep in seperators_Per:
        while " "+sep in text:
            text=text.replace(" "+sep,sep)
    text=text.replace(":",": ")
    while " :" in text:
        text=text.replace(" :",":")
    while "  " in text:
        text=text.replace("  "," ")
    while "  " in text:
        text=text.replace("  "," ")
    for i1 in numbers:
        for i2 in numbers:
            text=text.replace(str(i1)+" -"+str(i2),str(i1)+"-"+str(i2))
            text=text.replace(str(i1)+"- "+str(i2),str(i1)+"-"+str(i2))
    text=text.replace("ﻁ","ط")
    text=text.replace("ﺒ","ب")
    text=text.replace("ﺎ","ا")
    text=text.replace("ﻲ","ي")
    text=text.replace("ﻰ","ي")
    text=text.replace("ﺌ","ي")
    text=text.replace("ﯽ","ي")
    text=text.replace("ﻯ","ي")
    text=text.replace("ﻱ","ي")
    text=text.replace("ﻰ","ي")
    text=text.replace("ﺊ","ي")
    text=text.replace("ﻲ","ي")
    text=text.replace("ﻳ","ي")
    text=text.replace("أ","ا")
    text=text.replace("أ","ا")
    text=text.replace("إ","ا")
    text=text.replace("آ","ا")
    text=text.replace("ا","ا")
    text=text.replace("آ","ا")
    text=text.replace("أ","ا")
    text=text.replace("إ","ا")
```

```
789.  text=text.replace("ٱ","ا")
790.  text=text.replace("ﺐ","ب")
791.  text=text.replace("ﺑ","ب")
792.  text=text.replace("ﭗ","پ")
793.  text=text.replace("ﺒ","ب")
794.  text=text.replace("ﭙ","پ")
795.  text=text.replace("ۀ","ه")
796.  text=text.replace("ة","ه")
797.  text=text.replace("ﻩ","ه")
798.  text=text.replace("ﻪ","ه")
799.  text=text.replace("ﻫ","ه")
800.  text=text.replace("ﻬ","ه")
801.  text=text.replace("ۂ","ه")
802.  text=text.replace("ﻫ","ه")
803.  text=text.replace("ھ","ه")
804.  text=text.replace("ﺖ","ت")
805.  text=text.replace("ﺗ","ت")
806.  text=text.replace("ﺘ","ت")
807.  text=text.replace("ﺕ","ت")
808.  text=text.replace("ﺚ","ث")
809.  text=text.replace("ﺟ","ج")
810.  text=text.replace("ﭽ","چ")
811.  text=text.replace("ﺣ","ح")
812.  text=text.replace("ﺤ","ح")
813.  text=text.replace("ﺧ","خ")
814.  text=text.replace("ﺪ","د")
815.  text=text.replace("ﮈ","د")
816.  text=text.replace("ﺩ","د")
817.  text=text.replace("ﺫ","د")
818.  text=text.replace("ﮊ","د")
819.  text=text.replace("ﺬ","ذ")
820.  text=text.replace("ﺭ","ر")
821.  text=text.replace("ﻲ","ي")
822.  text=text.replace("ﻣ","م")
823.  text=text.replace("ﺣ","ح")
824.  text=text.replace("ﻤ","م")
825.  text=text.replace("ﺪ","د")
826.  text=text.replace("ﺿ","ض")
827.  text=text.replace("ﻋ","ع")
828.  text=text.replace("ﺪ","د")
829.  text=text.replace("ﺭ","ر")
830.  text=text.replace("ﺳ","س")
831.  text=text.replace("ﻭ","و")
832.  text=text.replace("ﻟ","ل")
833.  text=text.replace("ﻛ","ک")
834.  text=text.replace("ﻧ","ن")
835.  text=text.replace("ﺣ","ح")
836.  text=text.replace("ﻏ","غ")
837.  text=text.replace("ﻓ","ف")
838.  text=text.replace("ﻋ","ع")
839.  text=text.replace("ﻓ","ف")
840.  text=text.replace("ﺕ","ت")
841.  text=text.replace("ﻋ","ع")
842.  text=text.replace("ﻲ","ي")
843.  text=text.replace("ﺟ","ج")
844.  text=text.replace("ﻳ","ي")
845.  text=text.replace("ﻯ","ي")
846.  text=text.replace("۱","1")
847.  text=text.replace("ء","")
848.  text=text.replace("۲","2")
849.  text=text.replace("۳","3")
```

```
    text=text.replace("٤","4")
    text=text.replace("٥","5")
    text=text.replace("٨","8")
    text=text.replace("٧","7")
    text=text.replace("٦","6")
    text=text.replace("ي","ي")
    text=text.replace("و","و")
    text=text.replace("\ufeff","")
    text=text.replace("ي","ي")
    text=text.replace("5","5")
    text=text.replace("¬","-")
    text=text.replace(" ،","،")
    text=text.replace(" .",".")
    text=text.replace("'","")
    text=text.replace("  "," ")
    text=text.replace("ﺗ","ت")
    text=text.replace("-","")
    text=text.replace("ﻪ","ه")
    text=text.replace("ﺘ","ت")
    text=text.replace("ﮔ","گ")
    text=text.replace("ﻟ","ل")
    text=text.replace("ﺷ","ش")
    text=text.replace("ﻫ","ه")
    text=text.replace("ﻧ","ن")
    text=text.replace("ﺧ","خ")
    text=text.replace("ﻃ","ط")
    text=text.replace("م","م")
    text=text.replace("ي","ي")
    text=text.replace("س","س")
    text=text.replace("-","-")
    text=text.replace(" هاي "," هاي")
    text=text.replace(" ها "," ها")
    text=text.replace(" ي "," ي")
    text=text.replace(" اي "," اي")
    text=text.replace("–","-")
    text=text.replace("ﻻ","لا")
    text=text.replace(" -","-")
    text=text.replace("- ","-")
    text=text.replace("دانشكده","دانشکده")
    text=text.replace("ترجمهي","ترجمه")
    text=text.replace("با مقدمه","بامقدمه")
    text=text.replace("به كوشش","به‌کوشش")
    text=text.replace("پژوهش نامه","پژوهش‌نامه")
    text=text.replace("فصل نامه","فصل‌نامه")
    text=text.replace("فصلنامه","فصل‌نامه")
    text=text.replace("بين المللي","بین‌المللی")
    text=text.replace("پايان نامه","پایان‌نامه")
    text=text.replace("پايان نامه","پایان‌نامه")
    text=text.replace("پايان‌نامهي","پایان‌نامه")
    text=text.replace("رسالهٔ","رساله")
    text=text.replace("رسالهي","رساله")
    text=text.replace("رسالهي","رساله")
    text=text.replace("مجلهي","مجله")
    text=text.replace("مجلهي","مجله")
    text=text.replace("دورهي","دوره")
    text=text.replace("شمارهي","شماره")
    text=text.replace("دورهي","دوره")
    text=text.replace("شمارهي","شماره")
    text=dash_editor(text)
    return text
def Virayesh_Eng(text):
```

```
911.    text=convertnumbs(text)
912.    for c in engalph+ENGALPH:
913.        text=text.replace(c+"-",c+" ")
914.        text=text.replace("-"+c," "+c)
915.    for n in numbers:
916.        text=text.replace(str(n)+" th ",str(n)+"th ")
917.        text=text.replace(str(n)+" -",str(n)+"-")
918.        text=text.replace("- "+str(n),"-"+str(n))
919.    text=text.replace(str(2)+" nd ",str(2)+"nd ")
920.    text=text.replace(str(1)+" st ",str(1)+"st ")
921.    text=text.replace(" "," ")
922.    text=text.replace("procedings of","proceedingsof")
923.    text=text.replace("proceding of","procedingof")
924.    text=text.replace("proceedings of","proceedingsof")
925.    text=text.replace("proceeding of","proceedingof")
926.    text=text.replace("Proceedings of","proceedingsof")
927.    text=text.replace("Proceeding of","proceedingof")
928.    text=text.replace("new york","newyork")
929.    text=text.replace("New york","newyork")
930.    text=text.replace("New York","newyork")
931.    text=text.replace("new York","newyork")
932.    text=text.replace(" "," ")
933.    text=text.replace("”"," ")
934.    text=text.replace("”"," ")
935.    text=text.replace(" "," ")
936.    text=text.replace(" "," ")
937.    text=text.replace("et al","etal")
938.    text=text.replace("–","-")
939.    text=text.replace("-","-")
940.    text=text.replace(" "," ")
941.    text=text.replace("“"," ")
942.    text=text.replace("”"," ")
943.    text=text.replace("&"," & ")
944.    text=text.replace("m.sc","msc")
945.    text=text.replace("ph.d","phd")
946.    text=text.replace("M.sc.","msc")
947.    text=text.replace("Ph.d.","phd")
948.    text=text.replace("M.Sc.","msc")
949.    text=text.replace("M. Sc.","msc")
950.    text=text.replace("Ph.D.","phd")
951.    text=text.replace("Ph. D.","phd")
952.    text=text.replace("M.SC.","msc")
953.    text=text.replace("PH.D","phd")
954.    text=sep_text_from_number(text)
955.    text=text.replace("Rout ledge","routledge")
956.    text=text.replace("Routledge","routledge")
957.    for sep in seperators_Eng:
958.        text=text.replace(sep,sep+" ")
959.    for sep in seperators_Eng:
960.        while " "+sep in text:
961.            text=text.replace(" "+sep,sep)
962.    text=text.replace(":",": ")
963.    while " :" in text:
964.        text=text.replace(" :",":")
965.    while "  " in text:
966.        text=text.replace("  "," ")
967.    while "  " in text:
968.        text=text.replace("  "," ")
969.    for i1 in numbers:
970.        for i2 in numbers:
971.            text=text.replace(str(i1)+" -"+str(i2),str(i1)+"-"+str(i2))
```

```
            text=text.replace(str(i1)+"- "+str(i2),str(i1)+"-"+str(i2))
    return text
def is_a_written_number1(token):
    if token in ["1st","2nd","3rd"]:
        return True
    for i in list(range(4,100)):
        if token==str(i)+"th":
            return True
    return False
def is_a_written_number2(token):
    one_dig=["first","second","third","fourth","fifth","sixth","seventh","eighth","ninth"]
    spec=["tenth","eleventh","twelfth","thirteenth","fourteenth","fifteenth","sixteenth","seventeenth","eighteenth","
nineteenth","twentieth",
          "thirtieth","fortieth","fiftieth","sixtieth","seventieth","eightieth","ninetieth"]
    tenth=["twenty","thirty","forty","fifty","sixty","seventy","eighty","ninety"]
    nums=[]
    for num1 in tenth:
        for num2 in one_dig:
            nums.append(num1+num2)
            nums.append(num1+"-"+num2)
    if token in one_dig+spec+nums:
        return True
    return False
def is_a_written_number(token):
    yekan=["یک","دو","سه","چهار","پنج","شش","هفت","هشت","نه"]
    yek=["یک","اول","دوم","سوم","چهارم","پنجم","ششم","هفتم","هشتم","نهم","دهم","یازدهم","دوازدهم","سیزدهم","چهاردهم","پانزدهم","
شانزدهم","هفدهم","هجدهم",
         "نوزدهم","بیستم"]
    yekomin=[]
    for y in yek:
                yekomin.append(y+"ین")
            specials=["ده","یازده","دوازده","سیزده","چهارده","پانزده","شانزده",
                      "هفده","هجده","نوزده"]
            dahgan=["بیست","سي","چهل","پنجاه","شصت","هفتاد","هشتاد","نود"]
            if token in yekan+specials+dahgan+yek+yekomin:
                return True
            else:
                return False

        def is_mon_seas_name_Per(token):
            months1=["فروردین","اردیبهشت","خرداد","تیر","مرداد","شهریور",
                     "مهر","آبان","ابان","آذر","اذر","دي","بهمن","اسفند"]
            months2=["فروردین‌ماه","اردیبهشت‌ماه","خردادماه","تیرماه","مردادماه","شهریورماه",
                     "مهرماه","آبان‌ماه","ابان‌ماه","آذرماه","اذرماه","دي‌ماه","بهمن‌ماه","اسفندماه"]
            seasons=["بهار","تابستان","پاییز","زمستان"]
            months3=[]
            for i in list(range(len(months1))):
                for j in list(range(i+1, len(months1))):
                    months3.append(months1[i]+""+months1[j])
            months=months1+months2+months3
            if token in months+seasons:
                return True
            else:
                return False
        def is_mon_seas_name_Eng(token):
            months=["january","february","march","april","may","june","july","august","september","october","no
vember","December","jan","feb","mar","apr","may","jun","jul","aug","sep","sept","oct","nov","dec"]
            seasons=["spring","summer","fall","winter"]
            if token in months+seasons:
                return True
            else:
```

```
1030.                 return False
1031.
1032.
1033.         def contains_volume_ind_Both(token):
1034.             tmp=token
1035.             indicators=["v","vol","volume","دوره","دوره‌ي","سال","س","د"]
1036.             while is_number(tmp[-1:]):
1037.                 tmp=tmp[:-1]
1038.             if tmp in indicators:
1039.                 return True
1040.             else:
1041.                 return False
1042.         def contains_num_ind_Both(token):
1043.             tmp=token
1044.             indicators=["issue","number","num","n","شماره","شماره‌ي","شماره‌هاي","ش"]
1045.             while is_number(tmp[-1:]):
1046.                 tmp=tmp[:-1]
1047.             if tmp in indicators:
1048.                 return True
1049.             else:
1050.                 return False
1051.         def contains_series_ind_Both(token):
1052.             tmp=token
1053.             indicators=["ed","edition","چاپ","چ","چ."]
1054.             while is_number(tmp[-1:]):
1055.                 tmp=tmp[:-1]
1056.             if tmp in indicators:
1057.                 return True
1058.             else:
1059.                 return False
1060.         def contains_parts_ind_Per(token):
1061.             tmp=token
1062.             indicators=["جلد","ج","ج."]
1063.             while is_number(tmp[-1:]):
1064.                 tmp=tmp[:-1]
1065.             if tmp in indicators:
1066.                 return True
1067.             else:
1068.                 return False
1069.
1070.         def contains_page_ind_Both(token):
1071.             indicators=["ص","صص","صفحات","صفحه","صفحه‌هاي","p","pp","page","pages"]
1072.             tmp=token
1073.             while is_number(tmp[-1:]) or tmp[-1:]=="-":
1074.                 tmp=tmp[:-1]
1075.             if tmp in indicators:
1076.                 return True
1077.             else:
1078.                 return False
1079.         def is_page_number_Both(token):
1080.             indicators=["p","pp","page","pages","ص","صص","صفحات","صفحه","صفحه‌هاي"]
1081.             numbers=['0','1','2','3','4','5','6','7','8','9']
1082.             tmp=token
1083.             if not hasNumbers(tmp):
1084.                 return False
1085.             if contains_page_ind_Both(tmp):
1086.                 while not is_number(tmp[:1]):
1087.                     tmp=tmp[1:]
1088.             pnum=list(tmp)
1089.             for pn in pnum:
1090.                 if not str(pn) in numbers+['']:
```

```
                return False
            if '-' in pnum:
                return True
            else:
                return False
        def sep_text_from_number(token):
            tmp=token
            inds=["ش","ش.","شماره","شماره‌ي","شماره‌هاي","د","د.","س","س.","سال","دوره‌ي","جلد","چاپ","ج","ج.","چ","چ.","دوره","ص","ص.","صص","صص.",
                  "صفحات","صفحه","صفحه‌هاي","صفحه هاي","فروردين","ارديبهشت","خرداد","تير","مرداد","شهريور","مهر","آبان","ابان","آذر","دي","بهمن",
                  "اسفند","بهار","تابستان","پاييز","زمستان","v","vol","volume","issue","number","num","n","ed","edition","p","pp","page","pages"]
            numbers=["0","1","2","3","4","5","6","7","8","9"]
            for ind in inds:
                for n in numbers:
                    tmp=tmp.replace(ind+n,ind+" "+n)
            return tmp
        def dash_editor(text):
            for n in numbers:
                text=text.replace(n+" -",n+"-")
                text=text.replace(n+"- ",n+"-")
            return text

        def is_number(s):
            try:
                int(s)
                return True
            except ValueError:
                return False
        def hasNumbers(inputString):
                return any(char.isdigit() for char in inputString)
        def is_year_number(token):
            yearindicators=["ش","ق","م"]
            temp=token.replace("(","")
            temp=temp.replace(")","")
            if is_number(temp):
                if 1000<int(temp)<2020:
                    return True
                else:
                    return False
            elif is_number(temp[:-1]) and token[-1:] in yearindicators:
                if 1000<int(temp[:-1])<2020:
                    return True
                else:
                    return False
            else:
                return False

        def contain_l_name_indicator(token,lnameindicators):
            if token in lnameindicators:
                return True
            else:
                return False

        def special_year_editting(ref):
            ref=ref.replace("  "," ")
            numbers=[0,1,2,3,4,5,6,7,8,9]
            yearindicators=["ش","ق","م"]
            for n in numbers:
                for c in yearindicators:
```

```
1149.                 ref=ref.replace(str(n)+" "+c,str(n)+c)
1150.             return ref
1151.
1152.         def convertnumbs(ref):
1153.             newref=""
1154.             for c in ref:
1155.                 if c=="۱":
1156.                     newref=newref+unidecode(str(1))
1157.                 elif c=="۲":
1158.                     newref=newref+unidecode(str(2))
1159.                 elif c=="۳":
1160.                     newref=newref+unidecode(str(3))
1161.                 elif c=="۴":
1162.                     newref=newref+unidecode(str(4))
1163.                 elif c=="۵":
1164.                     newref=newref+unidecode(str(5))
1165.                 elif c=="۶":
1166.                     newref=newref+unidecode(str(6))
1167.                 elif c=="۷":
1168.                     newref=newref+unidecode(str(7))
1169.                 elif c=="۸":
1170.                     newref=newref+unidecode(str(8))
1171.                 elif c=="۹":
1172.                     newref=newref+unidecode(str(9))
1173.                 elif c=="۰":
1174.                     newref=newref+unidecode(str(0))
1175.                 else:
1176.                     newref=newref+c
1177.             return newref
1178.
1179.
1180.         def replace_sep_with_space(tmp,seperators):
1181.             for s in seperators:
1182.                 tmp=tmp.replace(str(s), " ")
1183.             if not is_page_number_Both(tmp):
1184.                 tmp=tmp.replace("-"," ")
1185.             return tmp
1186.         def is_ended_with_seperator(token,seperators):
1187.             if isendedwith(token,seperators):
1188.                 return True
1189.             else:
1190.                 return False
1191.
1192.         def isendedwith(temp, somelist):
1193.             for item in somelist:
1194.                 if temp.endswith(item):
1195.                     return True
1196.             return False
1197.
1198.         def Extract_features_Both(par):
1199.             par=Virayesh_Both(par)
1200.             fvector=[]
1201.             flag=False
1202.             num_words=0
1203.             num_seperators=0
1204.             num_numbers=0
1205.             num_years=0
1206.             flag_page_indicator=False
1207.             flag_vol_indicator=False
1208.             flag_issue_indicator=False
1209.             flag_edition_indicator=False
```

```
1210.             flag_part_indicator=False
1211.             num_dots=0
1212.             num_alph=0
1213.             num_uni_indicators=0
1214.             num_trans_indicators=0
1215.             num_thesis_indicators=0
1216.             num_pub_indicators=0
1217.             num_cities=0
1218.             flag_journal_name_indicator=False
1219.             flag_conf_name_indicator=False
1220.             if par.startswith("[") or par.startswith("["):
1221.                 fvector.append(int(True))
1222.             else:
1223.                 fvector.append(int(False))
1224.             chars=list(par)
1225.             if len(chars)<100:
1226.                 par70=par
1227.             else:
1228.                 par70=par[:100]
1229.             chars70=list(par70)
1230.             for ch in chars70:
1231.                 if ch in seperators_Both+["-"]:
1232.                     num_seperators+=1
1233.                 if ch in ["."]:
1234.                     num_dots+=1
1235.                 if ch in peralph+engalph+ENGALPH+numbers:
1236.                     num_alph+=1
1237.             tmp=replace_sep_with_space(par,seperators_Both)
1238.             tmp=tmp.replace("  "," ")
1239.             tmp=tmp[3:]
1240.             tokens=tmp.split(" ")
1241.             for token in tokens:
1242.                 if hasNumbers(token):
1243.                     num_numbers+=1
1244.                 if is_year_number(token):
1245.                     num_years+=1
1246.                 if token in ["ص","صص","صفحات","صفحه","صفحه‌هاي","p","pp","page","pages"]:
1247.                     flag_page_indicator=True
1248.                 if token in ["دوره","دوره‌ي","سال","س","د","v","vol","volume"]:
1249.                     flag_vol_indicator=True
1250.                 if token in ["شماره","شماره‌ي","شماره‌هاي","ش","issue","number","num","n"]:
1251.                     flag_issue_indicator=True
1252.                 if token in ["چاپ","چ","ed","edition"]:
1253.                     flag_edition_indicator=True
1254.                 if token in ["جلد","ج"]:
1255.                     flag_part_indicator=True
1256.                 if token in ["دانشگاه","پژوهشگاه","پژوهشکده","دانشکده","پیام","نور","آزاد","گروه","استان","ایران","شهید","university",
    "school"
1257.                         "faculty","center","department"]:
1258.                     num_uni_indicators+=1
1259.                 if token in ["ترجمه","ترجمه‌ي","مترجم","به‌کوشش","بامقدمه","بامقدمه‌ي","تصحیح","ویراستار"]:
1260.                     num_trans_indicators+=1
1261.                 if token in ["استاد","راهنما","پایان‌نامه","رساله","کارشناسي","کارشناسي‌ارشد","دکتري","دکترا","رشته","رساله‌ي","راهنمايي
    ",
1262.                         "thesis","dissertation","phd","master","msc","degree","doctor","degree","doctoral","ma","
    philosophy","phd's","master's","unpublished"]:
1263.                     num_thesis_indicators+=1
1264.                 if token in ["انتشارات","نشر","ناشر","الانتشار","چاپ","جهاد","pub","inc","press","publishing","publication","
    publications"]:
1265.                     num_pub_indicators+=1
1266.                 if token in ["قم","نجف","بیروت","یزد","تبریز","شیراز","اصفهان","تهران",
```

```
                    "newyork","london","united","states","us","uk","britain","Melbourne","california"]:
                num_cities+=1
            if token in ["فصلنامه","پژوهشنامه","ماهنامه","دوماهنامه","نشریه","دوفصلنامه","مجله","پژوهش","تحقیقات",
                    "تازه‌هاي","مطالعات","journal","j","research","international","int","letters","let","science","scien
                flag_journal_name_indicator=True
            if token in ["همایش","کنفرانس","سمینار","کنگره","بین‌المللي","ملي","ایران","انجمن","اولین","نخستین","یکمین","دومین",
                    "هفتمین","ششمین","پنجمین","چهارمین","سومین
                    "هفدهمین","شانزدهمین","پانزدهمین","چهاردهمین","سیزدهمین","دوازدهمین","یازدهمین","دهمین","نهمین","هشتمین
                    "مقالات","خلاصه","نوزدهمین","هجدهمین
                    "international","int","conference","symposium","congress","proceedings","annual","semin
                flag_con_name_indicator=True
        fvector.append(len(tokens))
        fvector.append(num_seperators)
        fvector.append(float(num_numbers/(num_words+1)))
        fvector.append(num_years)
        fvector.append(int(flag_page_indicator))
        fvector.append(int(flag_vol_indicator))
        fvector.append(int(flag_issue_indicator))
        fvector.append(int(flag_edition_indicator))
        fvector.append(int(flag_part_indicator))
        fvector.append(num_uni_indicators)
        fvector.append(num_trans_indicators)
        fvector.append(num_thesis_indicators)
        fvector.append(num_pub_indicators)
        fvector.append(num_cities)
        fvector.append(int(flag_journal_name_indicator))
        fvector.append(int(flag_conf_name_indicator))
        return fvector

    def Extract_features_type_Eng(par):
        par=Virayesh_Eng(par)
        par=par.lower()
        fvector=[]
        fvector.append(int(":" in par))
        num_numbers=0
        flag_page_indicator=False
        flag_page_number=False
        flag_vol_indicator=False
        flag_issue_indicator=False
        flag_edition_indicator=False
        flag_month=False
        num_uni_indicators=0
        num_uni_2_indicators=0
        flag_in_eds=False
        num_thesis_indicators=0
        num_pub_indicators=0
        num_cities=0
        flag_journal_name_indicator=False
        flag_journal=False
        num_conf_name_indicator=0
        flag_book_indicator=False
        flag_int=False
        flag_written_number1=False
        flag_written_number2=False
        flag_doi=False
        tmp=replace_sep_with_space(par,seperators_Eng)
        tmp=tmp.replace("  "," ")
        tokens=tmp.split(" ")
```

```
1323.             for token in tokens:
1324.                 if hasNumbers(token):
1325.                     num_numbers+=1
1326.                 if contains_page_ind_Both(token):
1327.                     flag_page_indicator=True
1328.                 if is_page_number_Both(token):
1329.                     flag_page_number=True
1330.                 if contains_volume_ind_Both(token):
1331.                     flag_vol_indicator=True
1332.                 if contains_num_ind_Both(token):
1333.                     flag_issue_indicator=True
1334.                 if contains_series_ind_Both(token):
1335.                     flag_edition_indicator=True
1336.                 if is_mon_seas_name_Eng(token):
1337.                     flag_month=True
1338.                 if token in ["university","school","college"]:
1339.                     num_uni_indicators+=1
1340.                 if token in ["faculty","center","department"]:
1341.                     num_uni_2_indicators+=1
1342.                 if token in ["eds"]:
1343.                     flag_in_eds=True
1344.                 if is_a_written_number1(token):
1345.                     flag_written_number1=True
1346.                 if is_a_written_number2(token):
1347.                     flag_written_number2=True
1348.                 if token in ["thesis","dissertation","phd","master","msc","degree","doctor","degree","doctoral","m
    a","philosophy","phd's","master's","unpublished"]+["mphil","diss","ms"]:
1349.                     num_thesis_indicators+=1
1350.                 if is_in_pn_Eng(token,pnames_Eng):
1351.                     num_pub_indicators+=1
1352.                 if token in ["newyork","london","united","states","us","uk","britain","melbourne"]+["canada","austr
    alia","iran","tehran","egypt",
1353.                                                                 "england","sweden","malaysia","mashhad",
    "shiraz",
1354.                                                                 "tabriz","yazd","esfahan","nj"]:
1355.                     num_cities+=1
1356.                 if token in ["journal"]:
1357.                     flag_journal=True
1358.                 if token in ["j","research","letters","lett","science","sciences","sci","review","advances","advance
    d","adv","ieee","transactions"
1359.                                             "bulletin"]:
1360.                     flag_journal_name_indicator=True
1361.                 if token in ["conference","conf","symposium","symp","congress","proceedingsof","proceedingof",
    "proc","workshop","presented","meeting","paper"]+["econference","papers","working"]:
1362.                     num_conf_name_indicator+=1
1363.                 if token in ["handbook","pub","inc","press","publishing","publication","publications","edition","pub
    lication"]+["lncs","routledge","lecture","isbn"]:
1364.                     flag_book_ind=True
1365.                 if token in ["international","int"]:
1366.                     flag_int=True
1367.                 if token in ["doi"]:
1368.                     flag_doi=True
1369.             fvector.append(num_numbers)
1370.             fvector.append(int(flag_page_indicator))
1371.             fvector.append(int(flag_page_number))
1372.             fvector.append(int(flag_vol_indicator))
1373.             fvector.append(int(flag_issue_indicator))
1374.             fvector.append(int(flag_edition_indicator))
1375.             fvector.append(int(flag_month))
1376.             fvector.append(num_uni_indicators)
1377.             fvector.append(num_uni_2_indicators)
```

```
1378.             fvector.append(flag_in_eds)
1379.             fvector.append(flag_written_number1)
1380.             fvector.append(flag_written_number2)
1381.             fvector.append(num_thesis_indicators)
1382.             fvector.append(int(num_pub_indicators))
1383.             fvector.append(int(num_cities))
1384.             fvector.append(int(flag_journal_name_indicator))
1385.             fvector.append(int(flag_journal))
1386.             fvector.append(int(num_conf_name_indicator))
1387.             fvector.append(int(flag_book_indicator))
1388.             fvector.append(int(flag_int))
1389.             fvector.append(int(flag_doi))
1390.             return fvector
1391.
1392.         def Extract_features_type_Per(par):
1393.             par=Virayesh_Per(par)
1394.             fvector=[]
1395.             fvector.append(int(":" in par))
1396.             num_numbers=0
1397.             flag_page_indicator=False
1398.             flag_page_number=False
1399.             flag_vol_indicator=False
1400.             flag_issue_indicator=False
1401.             flag_edition_indicator=False
1402.             flag_part_indicator=False
1403.             flag_month=False
1404.             num_uni_indicators=0
1405.             num_trans_indicators=0
1406.             num_thesis_indicators=0
1407.             flag_pub_indicators=False
1408.             flag_cities=False
1409.             flag_journal_name_indicator=False
1410.             flag_conf_name_indicator=False
1411.             tmp=replace_sep_with_space(par,seperators_Per)
1412.             tmp=tmp.replace("  "," ")
1413.             tokens=tmp.split(" ")
1414.             for token in tokens:
1415.                 if hasNumbers(token):
1416.                     num_numbers+=1
1417.                 if contains_page_ind_Both(token):
1418.                     flag_page_indicator=True
1419.                 if is_page_number_Both(token):
1420.                     flag_page_number=True
1421.                 if contains_volume_ind_Both(token):
1422.                     flag_vol_indicator=True
1423.                 if contains_num_ind_Both(token):
1424.                     flag_issue_indicator=True
1425.                 if contains_series_ind_Both(token):
1426.                     flag_edition_indicator=True
1427.                 if contains_parts_ind_Per(token):
1428.                     flag_part_indicator=True
1429.                 if is_mon_seas_name_Per(token):
1430.                     flag_month=True
1431.                 if token in ["شهید","ایران","استان","گروه","آزاد","نور","پیام","پژوهشکده","دانشکده","پژوهشگاه","دانشگاه"]:
1432.                     num_uni_indicators+=1
1433.                 if token in ["تصحیح","بامقدمه","بامقدمه‌ي","به‌کوشش","مترجم","ترجمه‌ي","ترجمه"]:
1434.                     num_trans_indicators+=1
1435.                 if token in ["استاد","راهنما","پایان‌نامه","رساله","کارشناسي","کارشناسي‌ارشد","دکتري","دکترا","رشته","رساله‌ي","راهنمايي
    "]:
1436.                     num_thesis_indicators+=1
1437.                 if token in ["انتشارات","نشر","ناشر","الانتشار","چاپ","جهاد"]:
```

```
                flag_pub_indicators=True
            if token in ["تهران","اصفهان","شیراز","تبریز","یزد","بیروت","نجف","قم"]:
                flag_cities=True
            if token in ["فصلنامه","پژوهشنامه","ماهنامه","دوماهنامه","نشریه","دوفصلنامه","مجله","پژوهشنامه","فصل‌نامه","
    دوفصل‌نامه","ماهنامه","پژوهش","تحقیقات",
                        "تازه‌هاي","مطالعات"]:
                flag_journal_name_indicator=True
            if token in ["همایش","کنفرانس","سمینار","کنگره","بین‌المللي","ملي","ایران","انجمن","اولین","نخستین","یکمین","دومین","
    سومین","چهارمین","پنجمین","ششمین","هفتمین",
                        "هشتمین","نهمین","دهمین","یازدهمین","دوازدهمین","سیزدهمین","چهاردهمین","پانزدهمین","شانزدهمین","هفدهمین","
    هجدهمین","نوزدهمین","خلاصه","مقالات"]:
                flag_con_name_indicator=True
        fvector.append(num_numbers)
        fvector.append(int(flag_page_indicator))
        fvector.append(int(flag_vol_indicator))
        fvector.append(int(flag_issue_indicator))
        fvector.append(int(flag_edition_indicator))
        fvector.append(int(flag_part_indicator))
        fvector.append(int(flag_month))
        fvector.append(num_uni_indicators)
        fvector.append(num_trans_indicators)
        fvector.append(num_thesis_indicators)
        fvector.append(int(flag_pub_indicators))
        fvector.append(int(flag_cities))
        fvector.append(int(flag_journal_name_indicator))
        fvector.append(int(flag_conf_name_indicator))
        return fvector

    def Extract_features_Parse_Eng(ref):
        text=ref.text
        text=Virayesh_Eng(text)
        try:
            while not text[:1] in engalph+ENGALPH:
                text=text[1:]
        except:
            pass
        tokens=text.split(" ")
        ttext=""
        for i in list(range(len(tokens))):
            tmp=tokens[i][:-1]
            if not is_f_name_indicator(tmp.lower(),Fnameindicators_Eng):
                tmp=replace_sep_with_space(tmp,seperators_Eng)
            while tmp.startswith(" "):
                tmp=tmp[1:]
            while tmp.endswith(" "):
                tmp=tmp[:-1]
            ttext=ttext+tmp+tokens[i][-1:]+" "
        ttext=ttext.replace("  "," ")
        tokens=ttext.split(" ")
        currindex=1
        fvectors=[]
        numnumbers=0
        for i in list(range(len(tokens)-1)):
            token=tokens[i]
            if token in ["\r\n","\r","\n"]:
                continue
            previousindex=0
            fvector=[]
            fvector.append(currindex)
            fvector.append(int(is_ended_with_seperator(token,seperators_Eng)))
```

```
1495.                 if is_ended_with_seperator(token,seperators_Eng) and not is_f_name_indicator(token.lower(),
      Fnameindicators_Eng):
1496.                     token=token[:-1]
1497.                 else:
1498.                     flag=False
1499.                 fvector.append(int(token.lower() in ["in"]))
1500.                 fvector.append(int(token.lower() in ["in:"]))
1501.                 fvector.append(int(token.lower() in ["and"]))
1502.                 fvector.append(int(token.lower() in ["&"]))
1503.                 fvector.append(int(token.lower() in ["or"]))
1504.                 fvector.append(int(token.lower() in ["of"]))
1505.                 fvector.append(int(token.lower() in ["on"]))
1506.                 fvector.append(int(token.lower() in ["at"]))
1507.                 fvector.append(int(token.lower() in ["paper","presented","at","procedingsof","procedingof"]))
1508.                 fvector.append(int(token.lower() in ["ed","eds","editors","edited"]))
1509.                 currindex=currindex+1
1510.                 fvector.append(int(token[:1].isupper()))
1511.                 token=token.lower()
1512.                 fvector.append(int(token[-1:]==":"))
1513.                 if token[-1:]==":":
1514.                     token=token[:-1]
1515.                 fvector.append(int(contains_page_ind_Both(token)))
1516.                 fvector.append(int(is_page_number_Both(token)))
1517.                 fvector.append(int(is_number(token) or is_number(token[:-1])))
1518.                 if is_number(token) or is_number(token[:-1]):
1519.                     numnumbers=numnumbers+1
1520.                 fvector.append(numnumbers)
1521.                 fvector.append(int(is_year_number(token)))
1522.                 fvector.append(int(is_f_name_indicator(token.lower(),Fnameindicators_Eng)))
1523.                 fvector.append(int(is_contain_nonalphabetic(token, engalph+ENGALPH)))
1524.                 fvector.append(int(is_contain_num_alph(token,engalph+ENGALPH)))
1525.                 fvector.append(int(is_only_contain_num_alph(token,engalph+ENGALPH)))
1526.                 fvector.append(int(contains_volume_ind_Both(token)))
1527.                 fvector.append(int(contains_num_ind_Both(token)))
1528.                 fvector.append(int(contains_series_ind_Both(token)))
1529.                 fvector.append(int(is_mon_seas_name_Eng(token)))
1530.                 fvector.append(int(token in ["a"]))
1531.                 fvector.append(int(token in ["an"]))
1532.                 fvector.append(int(token in ["the"]))
1533.                 fvector.append(int(is_a_written_number1(token)))
1534.                 fvector.append(int(is_a_written_number2(token)))
1535.                 fvector.append(int(token in ["etal"]))
1536.                 fvector.append(int(token in ["university","school"]))
1537.                 fvector.append(int(token in ["faculty","center","department"]))
1538.                 fvector.append(int(token in ["thesis","dissertation","phd","master","msc","degree","doctor","degr
      ee","doctoral","ma","philosophy","phd's","master's","unpublished"]))
1539.                 fvector.append(int(token in ["pub","inc","press","publishing","publication","publications"]))
1540.                 fvector.append(int(is_in_pn_Eng(token,pnames_Eng)))
1541.                 fvector.append(int(token in ["newyork","london","united","states","us","uk","britain","Melbourne",
      "california"]))
1542.                 fvector.append(int(token in ["journal","j","research","letters","lett","science","sciences","sci","revi
      ew","advances","advanced","adv",
1543.                                            "bulletin"]))
1544.                 fvector.append(int(token in ["international","int"]))
1545.                 fvector.append(int(token in ["conference","symposium","congress","seminar"]))
1546.                 fvector.append(int(is_in_jn_Eng(token,jnames_Eng)))
1547.                 fvector.append(int(is_in_cn_Eng(token,cnames_Eng)))
1548.                 fvectors.append([token,fvector])
1549.             return fvectors
1550.         def Extract_features_Parse_Per(ref):
1551.             text=ref.text
```

```
1552.             text=Virayesh_Per(text)
1553.             try:
1554.                 while not text[0] in peralph:
1555.                     text=text[1:]
1556.             except:
1557.                 pass
1558.             tokens=text.split(" ")
1559.             ttext=""
1560.             for i in list(range(len(tokens))):
1561.                 tmp=tokens[i][:-1]
1562.                 if not is_f_name_indicator(tmp,Fnameindicators_Per):
1563.                     tmp=replace_sep_with_space(tmp,seperators_Per)
1564.                 if not hasNumbers(tmp):
1565.                     tmp=tmp.replace("-","")
1566.                 while tmp.startswith(" "):
1567.                     tmp=tmp[1:]
1568.                 while tmp.endswith(" "):
1569.                     tmp=tmp[:-1]
1570.                 ttext=ttext+tmp+tokens[i][-1:]+" "
1571.             ttext=ttext.replace("  "," ")
1572.             tokens=ttext.split(" ")
1573.             currindex=1
1574.             fvectors=[]
1575.             flag=False
1576.             numnumbers=0
1577.             for i in list(range(len(tokens)-1)):
1578.                 token=tokens[i]
1579.                 if token in ["\r\n","\r","\n"]:
1580.                     continue
1581.                 previousindex=0
1582.                 fvector=[]
1583.                 fvector.append(currindex)
1584.                 currindex=currindex+1
1585.                 fvector.append(int(flag))
1586.                 fvector.append(int(is_ended_with_seperator(token,seperators_Per)))
1587.                 if is_ended_with_seperator(token,seperators_Per) and not is_f_name_indicator(token,Fnamein
    dicators_Per):
1588.                     flag=True
1589.                     token=token[:-1]
1590.                 else:
1591.                     flag=False
1592.                 fvector.append(int(contains_page_ind_Both(token)))
1593.                 fvector.append(int(is_page_number_Both(token)))
1594.                 fvector.append(int(is_number(token) or is_number(token[:-1])))
1595.                 if is_number(token) or is_number(token[:-1]):
1596.                     numnumbers=numnumbers+1
1597.                 fvector.append(int(is_number(token) or is_number(token[:-1]))*numnumbers)
1598.                 fvector.append(int(is_year_number(token)))
1599.                 fvector.append(int(is_f_name_indicator(token,Fnameindicators_Per)))
1600.                 fvector.append(int(contain_l_name_indicator(token,lnameindicators)))
1601.                 fvector.append(int(is_contain_nonalphabetic(token, peralph)))
1602.                 fvector.append(int(is_contain_num_alph(token,peralph)))
1603.                 fvector.append(int(is_only_contain_num_alph(token,peralph)))
1604.                 fvector.append(int(contains_volume_ind_Both(token) or contains_num_ind_Both(token)))
1605.                 fvector.append(int(contains_series_ind_Both(token)))
1606.                 fvector.append(int(contains_parts_ind_Per(token)))
1607.                 fvector.append(int(is_mon_seas_name_Per(token)))
1608.                 fvector.append(int(is_a_written_number(token)))
1609.                 fvector.append(int(is_containing_Eng_alphabet(token)))
1610.                 fvector.append(int(is_et_al_Per(token)))
```

```
                fvector.append(int(token in ["دانشگاه","پژوهشگاه","دانشکده","پژوهشکده","پیام","نور","آزاد","گروه","استان","ایران","شهید"]))
                fvector.append(int(token in ["ترجمه","ترجمه‌ي","مترجم","به‌کوشش","بامقدمه","بامقدمه‌ي","تصحیح"] or token.startswith("ترجمه") or token.startswith("مترجم")))
                fvector.append(int(token in ["استاد","راهنما","پایان‌نامه","پایان‌نامه","رساله","کارشناسي","کارشناسي‌ارشد","دکتري","راهنمايي","رساله‌ي","رشته","دکترا"]))
                fvector.append(int(token in ["انتشارات","نشر","ناشر","الانتشار","چاپ","جهاد"]))
                fvector.append(int(token in ["تهران","اصفهان","شیراز","تبریز","یزد","بیروت","نجف","قم"]))
                fvector.append(int(is_j_ind_Per(token)))
                fvector.append(int(is_c_ind_Per(token)))
                fvector.append(int(token in ["و","تا", "الي"]))
                fvector.append(int(is_author_name_Per(token,names_Per)))
                fvector.append(format(j_name_score_Per(token,jnames_scores_Per),'04d'))
                fvector.append(format(c_name_score_Per(token,cnames_scores_Per),'04d'))
                fvector.append(format(title_name_score_Per(token,tw_scores_Per),'04d'))
                fvectors.append([token,fvector])
            return fvectors

        def Extract_features(par):
            text=Virayesh(par)
            text=replace_sep_with_sepspace(text,seperators)
            text=text.replace("  "," ")
            tokens=text.split(" ")
            fvectors=[]
            for i in list(range(len(tokens))):
                token=tokens[i]
                fvector=[]
                num_ch=0
                num_dot=0
                for c in list(token):
                    if c in peralph+engalph+ENGALPH:
                        num_ch+=1
                    if c in ["."]:
                        num_dot+=1
                if num_ch==0:
                    continue
                if num_ch>9:
                    num_ch=9
                fvector.append(int(i==0))
                fvector.append(num_ch)
                fvector.append(num_dot)
                fvector.append(int(token.lower() in ["and"]))
                fvector.append(int(token.lower() in ["&"]))
                fvector.append(int(token in ["و"]))
                fvector.append(int(token[:1].isupper()))
                token=token.lower()
                fvector.append(int(token[-1:] in ["."]))
                fvector.append(int(token[-1:] in [","]))
                fvector.append(int(token[-1:] in [";"]))
                fvector.append(int(token[-1:] in ["-"]))
                fvector.append(int(token[-1:] in ["،"]))
                fvector.append(int(token[-1:] in ["؛"]))
                fvector.append(int(is_f_name_indicator(token.lower(),Fnameindicators_Eng)))
                fvector.append(int(is_f_name_indicator(token,Fnameindicators_Per)))
                fvector.append(int(contain_l_name_indicator(token,lnameindicators)))
                fvector.append(int(token in seyed))
                fvector.append(int("etal" in token))
                fvector.append(int("همکاران" in token or "دیگران" in token))
                fvector.append(int(2))
                fvectors.append([token,fvector])
            return fvectors
```

```
1669.         def Extract_features_NameLabeling(par):
1670.             text=Virayesh(par)
1671.             text=replace_sep_with_sepspace(text,seperators)
1672.             text=text.replace("  "," ")
1673.             tokens=text.split(" ")
1674.             fvectors=[]
1675.             for i in list(range(len(tokens))):
1676.                 token=tokens[i]
1677.                 fvector=[]
1678.                 num_ch=0
1679.                 num_dot=0
1680.                 for c in list(token):
1681.                     if c in peralph+engalph+ENGALPH:
1682.                         num_ch+=1
1683.                     if c in ["."]:
1684.                         num_dot+=1
1685.                 if num_ch==0:
1686.                     continue
1687.                 if num_ch>9:
1688.                     num_ch=9
1689.                 fvector.append(int(i==0))
1690.                 fvector.append(num_ch)
1691.                 fvector.append(num_dot)
1692.                 fvector.append(int(token.lower() in ["and"]))
1693.                 fvector.append(int(token.lower() in ["&"]))
1694.                 fvector.append(int(token in ["و"]))
1695.                 fvector.append(int(token[:1].isupper()))
1696.                 token=token.lower()
1697.                 fvector.append(int(token[-1:] in ["."]))
1698.                 fvector.append(int(token[-1:] in [","]))
1699.                 fvector.append(int(token[-1:] in [";"]))
1700.                 fvector.append(int(token[-1:] in ["-"]))
1701.                 fvector.append(int(token[-1:] in ["،"]))
1702.                 fvector.append(int(token[-1:] in ["؛"]))
1703.                 fvector.append(int(is_f_name_indicator(token.lower(),Fnameindicators_Eng)))
1704.                 fvector.append(int(is_f_name_indicator(token,Fnameindicators_Per)))
1705.                 fvector.append(int(contain_l_name_indicator(token,lnameindicators)))
1706.                 fvector.append(int(token in seyed))
1707.                 fvector.append(int("etal" in token))
1708.                 fvector.append(int("همکاران" in token or "دیگران" in token))
1709.                 fvectors.append([token,fvector])
1710.             return fvectors
1711.         def ind_add(table):
1712.             inds=[]
1713.             for i in list(range(len(table))):
1714.                 if table[i][0]==1:
1715.                     rand=randint(0,9)
1716.                 inds.append(rand)
1717.             return inds
1718.
1719.         def ext_refs(doc):
1720.             text=doc.text
1721.             l_p=[]
1722.             strr=""
1723.             out_pars=[]
1724.             pars3=text.split("\n")
1725.             pars2=[]
1726.             for p in pars3:
1727.                 if not "-------" in p:
1728.                     pars2.append(p)
1729.             pars=[]
```

```
1730.             for par in pars2:
1731.                 chars=list(par)
1732.                 num_alphs=0
1733.                 for ch in chars:
1734.                     if ch in peralph+engalph+ENGALPH:
1735.                         num_alphs+=1
1736.                 if num_alphs>20 and (len(par.split(" "))>7) and (len(par.split(" "))<80):
1737.                     pars.append(par)
1738.             for i in list(range(len(pars))):
1739.                 par=pars[i]
1740.                 feats=Extract_features_Both(par)
1741.                 test_item=[]
1742.                 test_item.append(feats)
1743.                 df2=pd.DataFrame(test_item)
1744.                 x2=df2.drop(df2.columns[[]],axis=1)
1745.                 y_pred=svclassifier_ext_refs.predict(x2)
1746.                 if "....." in par:
1747.                     l_p.append(0)
1748.                 else:
1749.                     l_p.append(y_pred[0])
1750.                 out_pars.append(par)
1751.             mean_ind=find_ind(l_p)
1752.             n_l_p=edit_array(l_p,mean_ind)
1753.             for i in list(range(len(n_l_p))):
1754.                 if n_l_p[i]==1:
1755.                     strr=strr+str(n_l_p[i])+str(n_l_p[i])+str(n_l_p[i])+out_pars[i]+"\r\n"
1756.             return strr
1757.
1758.         def detect_type_Eng(text):
1759.             feats=Extract_features_type_Eng(text)
1760.             test_item=[]
1761.             test_item.append(feats)
1762.             df2=pd.DataFrame(test_item)
1763.             x2=df2.drop(df2.columns[[]],axis=1)
1764.             y_pred=svclassifier_Type_Eng.predict(x2)
1765.             if y_pred==1:
1766.                 return "Journal"
1767.             elif y_pred==2:
1768.                 return "Conference"
1769.             elif y_pred==3:
1770.                 return "Book"
1771.             elif y_pred==4:
1772.                 return "Thesis"
1773.             else:
1774.                 return "NNN"
1775.
1776.         def detect_type_Per(text):
1777.             feats=Extract_features_type_Per(text)
1778.             test_item=[]
1779.             test_item.append(feats)
1780.             df2=pd.DataFrame(test_item)
1781.             x2=df2.drop(df2.columns[[]],axis=1)
1782.             y_pred=svclassifier_Type_Per.predict(x2)
1783.             if y_pred==1:
1784.                 return "Journal"
1785.             elif y_pred==2:
1786.                 return "Conference"
1787.             elif y_pred==3:
1788.                 return "Book"
1789.             elif y_pred==4:
1790.                 return "Thesis"
```

```
1791.             else:
1792.                 return "NNN"
1793.
1794.         def add_neighbours_features_NameLabeling(table):
1795.             newtable=[]
1796.             for i in list(range(len(table))):
1797.                 new_row=[]
1798.                 new_row.append(i)
1799.                 if table[i][0]==1:
1800.                     temp=["0","0"]
1801.                 else:
1802.                     temp=[temp[1],table[i-1][len(table[0])-2]]
1803.                 for j in list(range(2)):
1804.                     new_row.append(temp[j])
1805.                 for j in list(range(len(table[0]))):
1806.                     new_row.append(table[i][j])
1807.                 if i<len(table)-1:
1808.                     if table[i+1][0]==0:
1809.                         for j in list(range(1,len(table[0])-2)):
1810.                             new_row.append(table[i+1][j])
1811.                     else:
1812.                         for j in list(range(1,len(table[0])-2)):
1813.                             new_row.append("0")
1814.                 else:
1815.                     for j in list(range(1,len(table[0])-2)):
1816.                         new_row.append("0")
1817.                 newtable.append(new_row)
1818.             return newtable
1819.
1820.         def add_neighbours_features_Per(table):
1821.             newtable=[]
1822.             for i in list(range(len(table))):
1823.                 new_row=[]
1824.                 if table[i][0]==1 or i==0:
1825.                     temp=["0","0","0"]
1826.                 else:
1827.                     temp=[temp[1],temp[2],table[i-1][len(table[0])-1]]
1828.                 for j in list(range(3)):
1829.                     new_row.append(temp[j])
1830.
1831.                 for j in list(range(len(table[0]))):
1832.                     new_row.append(table[i][j])
1833.
1834.                 if i<len(table)-1:
1835.                     if table[i+1][0]>table[i][0]:
1836.                         for j in list(range(1,len(table[0])-1)):
1837.                             new_row.append(table[i+1][j])
1838.                     else:
1839.                         for j in list(range(1,len(table[0])-1)):
1840.                             new_row.append("0")
1841.                 else:
1842.                     for j in list(range(1,len(table[0])-1)):
1843.                         new_row.append("0")
1844.                 newtable.append(new_row)
1845.             return newtable
1846.
1847.
1848.         def add_neighbours_features_Eng(table):
1849.             newtable=[]
1850.             for i in list(range(len(table))):
1851.                 new_row=[]
```

```
1852.                 if table[i][0]==1 or i==0:
1853.                     temp=["0","0","0"]
1854.                 else:
1855.                     temp=[temp[1],temp[2],table[i-1][len(table[0])-2]]
1856.                 for j in list(range(3)):
1857.                     new_row.append(temp[j])
1858.
1859.                 for j in list(range(len(table[0])-1)):
1860.                     new_row.append(table[i][j])
1861.
1862.                 if i<len(table)-1:
1863.                     if table[i+1][0]>table[i][0]:
1864.                         for j in list(range(2,len(table[0])-2)):
1865.                             new_row.append(table[i+1][j])
1866.                     else:
1867.                         for j in list(range(2,len(table[0])-2)):
1868.                             new_row.append("0")
1869.                 else:
1870.                     for j in list(range(2,len(table[0])-2)):
1871.                         new_row.append("0")
1872.                 newtable.append(new_row)
1873.             return newtable
1874.
1875.         def strfeatures_to_intfeatures(table):
1876.             Labels=["0","author","title","year","journal","volume","pages","publisher","address","bikhial","scho
      ol","booktitle","series","month"]
1877.             for i in list(range(len(table))):
1878.                 for j in list(range(len(table[i]))):
1879.                     if table[i][j] in Labels:
1880.                         table[i][j]=Labels.index(table[i][j])
1881.             return table
1882.         def most_frequent(List):
1883.             counter = 0
1884.             num = List[0]
1885.             for i in List:
1886.                 curr_frequency = List.count(i)
1887.                 if(curr_frequency> counter):
1888.                     counter = curr_frequency
1889.                     num = i
1890.             return num
1891.         def edit_lbls(out_tokens,out_lbls):
1892.             new_lbls=[]
1893.             temp=[]
1894.             k=0
1895.             for i in list(range(len(out_lbls))):
1896.                 #print(out_tokens[i-k])
1897.                 if out_lbls[i]=="*":
1898.                     k+=1
1899.                 if out_lbls[i] in [1,2,4,7,8,10,11] and not is_contain_nonalphabetic(out_tokens[i-
      k],peralph+engalph+ENGALPH):
1900.                     temp.append(out_lbls[i])
1901.                 else:
1902.                     if len(temp)>0:
1903.                         mf=most_frequent(temp)
1904.                         for j in list(range(len(temp))):
1905.                             new_lbls.append(mf)
1906.                         temp=[]
1907.                     if out_lbls[i]!="*":
1908.                         new_lbls.append(out_lbls[i])
1909.                 #print(new_lbls)
1910.             for i in list(range(1,len(new_lbls)-1)):
```

```
                if new_lbls[i-1]==1 and new_lbls[i+1]==1 and new_lbls[i]!=1:
                    new_lbls[i]=1
            for i in list(range(1,len(new_lbls)-2)):
                if new_lbls[i-
    1]==1 and new_lbls[i+2]==1 and (out_tokens[i] in ["و","and","&"] or out_tokens[i+1] in ["و","and","&"]):
                    new_lbls[i]=1
                    new_lbls[i+1]=1
            return new_lbls

        def parse_Per(ref):
            #print(ref.text)
            feats=Extract_features_Parse_Per(ref)
            tempstr=ref.text
            tempstr=Virayesh_Per(tempstr)
            Author=""
            Title=""
            Year=""
            Journal=""
            Volume=""
            Pages=""
            Publisher=""
            Address=""
            School=""
            Conference=""
            Seri=""
            Month=""
            temp=[0,0,0]
            for i in list(range(len(feats))):
                new_row=[]
                for j in list(range(3)):
                    new_row.append(temp[j])
                for j in list(range(len(feats[0][1]))):
                    new_row.append(feats[i][1][j])
                for j in list(range(1,len(feats[0][1]))):
                    if i<len(feats)-1:
                        new_row.append(feats[i+1][1][j])
                    else:
                        new_row.append(0)
                test_item=[]
                test_item.append(new_row)
                y_pred=svclassifier_Parse_Per.predict((test_item))
                temp=[temp[1],temp[2],y_pred[0]]
                token=feats[i][0]
                if y_pred[0]==1:
                    index=tempstr.find(feats[i][0])
                    ll=len(feats[i][0])
                    if tempstr[index+ll:index+ll+1] in [".",",","؛",":","،","؛","-"]:
                if tempstr[index+ll+1:index+ll+2] in [".",",","؛",":","،","؛","-"]:
                    token=tempstr[index:index+ll+2]
                else:
                     token=tempstr[index:index+ll+1]
        out_lbls.append(y_pred[0])
        out_tokens.append(token)
        index=tempstr.find(token)
        ll=len(token)
        tempstr=tempstr[index+ll:]
        while tempstr[:1]==" ":
            tempstr=tempstr[1:]
        if tempstr[:1] in [".",",","؛",":","،","؛","-"]:
            out_lbls.append("*")
        fin_lbls=edit_lbls(out_tokens,out_lbls)
```

```
1971.         for i in list(range(len(fin_lbls))):
1972.         if fin_lbls[i]==1:
1973.             Author=Author+ out_tokens[i]+" "
1974.         elif fin_lbls[i]==2:
1975.             Title=Title+ out_tokens[i]+" "
1976.         elif fin_lbls[i]==3:
1977.             Year=Year+ out_tokens[i]+" "
1978.         elif fin_lbls[i]==4:
1979.             Journal=Journal+ out_tokens[i]+" "
1980.         elif fin_lbls[i]==5:
1981.             Volume=Volume+ out_tokens[i]+" "
1982.         elif fin_lbls[i]==6:
1983.             Pages=Pages+ out_tokens[i]+" "
1984.         elif fin_lbls[i]==7:
1985.             Publisher=Publisher+ out_tokens[i]+" "
1986.         elif fin_lbls[i]==8:
1987.             Address=Address+ out_tokens[i]+" "
1988.         elif fin_lbls[i]==10:
1989.             School=School+ out_tokens[i]+" "
1990.         elif fin_lbls[i]==11:
1991.             Conference=Conference+ out_tokens[i]+" "
1992.         elif fin_lbls[i]==12:
1993.             Seri=Seri+ out_tokens[i]+" "
1994.         elif fin_lbls[i]==13:
1995.             Month=Month+ out_tokens[i]+" "
1996.         auths_lbls=sep_auths_assign_lbls(Author)
1997.         authorss=sep_auths(auths_lbls)
1998.         if len(Year.split(" "))>1:
1999.             for y in Year.split(" "):
2000.                 try:
2001.                     if (int(y) or int(y[:-1])) and 3<len(list(y))<6:
2002.                         Year=y
2003.                         break
2004.                 except:
2005.                     pass
2006.             print(authorss)
2007.             print("Author:=  "+"-----".join(authorss))
2008.             print("Title:=  "+Title)
2009.             print("Year:=  "+Year)
2010.             print("Journal:=  "+Journal)
2011.             print("Volume:=  "+Volume)
2012.             print("Pages:=  "+Pages)
2013.             print("Publisher:=  "+Publisher)
2014.             print("Address:=  "+Address)
2015.             print("School:=  "+School)
2016.             print("Conference:=  "+Conference)
2017.             print("Seri:=  "+Seri)
2018.             print("Month:=  "+Month)
2019.             if ref.type=="Journal":
2020.                 new_ref=Reference(ref.lang,ref.type,authorss,Title,Year,Journal,Volume,Pages,Address,Seri,M
      onth,ref.text)
2021.             elif ref.type=="Conference":
2022.                 new_ref=Reference(ref.lang,ref.type,authorss,Title,Year,Conference,Volume,Pages,Address,Se
      ri,Month,ref.text)
2023.             elif ref.type=="Book":
2024.                 new_ref=Reference(ref.lang,ref.type,authorss,Title,Year,Publisher,Volume,Pages,Address,Seri,
      Month,ref.text)
2025.             elif ref.type=="Thesis":
2026.                 new_ref=Reference(ref.lang,ref.type,authorss,Title,Year,School,Volume,Pages,Address,Seri,M
      onth,ref.text)
2027.             else:
```

```
                new_ref=Reference(ref.lang,ref.type,authorss,Title,Year,School,Volume,Pages,Address,Seri,M
    onth,ref.text)
            return new_ref
        def parse_Eng(ref):
            tempstr=ref.text
            tempstr=Virayesh_Eng(tempstr)
            tempstr=tempstr.lower()
            feats=Extract_features_Parse_Eng(ref)
            Author=""
            Title=""
            Year=""
            Journal=""
            Volume=""
            Pages=""
            Publisher=""
            Address=""
            School=""
            Conference=""
            Seri=""
            Month=""
            temp=[0,0,0]
            for i in list(range(len(feats))):
                new_row=[]
                for j in list(range(3)):
                    new_row.append(temp[j])
                for j in list(range(len(feats[0][1]))):
                    new_row.append(feats[i][1][j])
                for j in list(range(1,len(feats[0][1]))):
                    if i<len(feats)-1:
                        new_row.append(feats[i+1][1][j])
                    else:
                        new_row.append(0)
                test_item=[]
                test_item.append(new_row)
                y_pred=svclassifier_Parse_Eng.predict((test_item))
                temp=[temp[1],temp[2],y_pred]
                if y_pred[0]==1:
                    index=tempstr.find(feats[i][0])
                    ll=len(feats[i][0])
                    if tempstr[index+ll:index+ll+1] in [".",",","؛",";","،","؛","-"]:
                if tempstr[index+ll+1:index+ll+2] in [".",",","؛",";","،","؛","-"]:
                    token=tempstr[index:index+ll+2]
                else:
                    token=tempstr[index:index+ll+1]
        out_lbls.append(y_pred[0])
        out_tokens.append(token)
        index=tempstr.find(token)
        ll=len(token)
        tempstr=tempstr[index+ll:]
        while tempstr[:1]==" ":
            tempstr=tempstr[1:]
        if tempstr[:1] in [".",",","؛",";","،","؛","-"]:
            out_lbls.append("*")
        fin_lbls=edit_lbls(out_tokens,out_lbls)
        for i in list(range(len(fin_lbls))):
        if fin_lbls[i]==1:
            Author=Author+ out_tokens[i]+" "
        elif fin_lbls[i]==2:
            Title=Title+ out_tokens[i]+" "
        elif fin_lbls[i]==3:
            Year=Year+ out_tokens[i]+" "
```

```
2088.            elif fin_lbls[i]==4:
2089.                Journal=Journal+ out_tokens[i]+" "
2090.            elif fin_lbls[i]==5:
2091.                Volume=Volume+ out_tokens[i]+" "
2092.            elif fin_lbls[i]==6:
2093.                Pages=Pages+ out_tokens[i]+" "
2094.            elif fin_lbls[i]==7:
2095.                Publisher=Publisher+ out_tokens[i]+" "
2096.            elif fin_lbls[i]==8:
2097.                Address=Address+ out_tokens[i]+" "
2098.            elif fin_lbls[i]==10:
2099.                School=School+ out_tokens[i]+" "
2100.            elif fin_lbls[i]==11:
2101.                Conference=Conference+ out_tokens[i]+" "
2102.            elif fin_lbls[i]==12:
2103.                Seri=Seri+ out_tokens[i]+" "
2104.            elif fin_lbls[i]==13:
2105.                Month=Month+ out_tokens[i]+" "
2106.            auths_lbls=sep_auths_assign_lbls(Author)
2107.            authorss=sep_auths(auths_lbls)
2108.            if len(Year.split(" "))>1:
2109.                for y in Year.split(" "):
2110.                    try:
2111.                        if (int(y) or int(y[:-1])) and 3<len(list(y))<6:
2112.                            Year=y
2113.                            break
2114.                    except:
2115.                        pass
2116.                print(authorss)
2117.                print("Author:=  "+"-----".join(authorss))
2118.                print("Title:=  "+Title)
2119.                print("Year:=  "+Year)
2120.                print("Journal:=  "+Journal)
2121.                print("Volume:=  "+Volume)
2122.                print("Pages:=  "+Pages)
2123.                print("Publisher:=  "+Publisher)
2124.                print("Address:=  "+Address)
2125.                print("School:=  "+School)
2126.                print("Conference:=  "+Conference)
2127.                print("Seri:=  "+Seri)
2128.                print("Month:=  "+Month)
2129.                if len(Year.split(" "))>1:
2130.                    if Year.split(" ")[0]==Year.split(" ")[1]:
2131.                        Year=Year.split(" ")[0]
2132.                if ref.type=="Journal":
2133.                    new_ref=Reference(ref.lang,ref.type,authorss,Title,Year,Journal,Volume,Pages,Address,Seri,M
      onth,ref.text)
2134.                elif ref.type=="Conference":
2135.                    new_ref=Reference(ref.lang,ref.type,authorss,Title,Year,Conference,Volume,Pages,Address,Se
      ri,Month,ref.text)
2136.                elif ref.type=="Book":
2137.                    new_ref=Reference(ref.lang,ref.type,authorss,Title,Year,Publisher,Volume,Pages,Address,Seri,
      Month,ref.text)
2138.                elif ref.type=="Thesis":
2139.                    new_ref=Reference(ref.lang,ref.type,authorss,Title,Year,School,Volume,Pages,Address,Seri,M
      onth,ref.text)
2140.                else:
2141.                    new_ref=Reference(ref.lang,ref.type,authorss,Title,Year,School,Volume,Pages,Address,Seri,M
      onth,ref.text)
2142.                return new_ref
2143.
```

```
2144.         with codecs.open(files_dir+"\\Per-jn.txt", 'r',encoding='utf-8') as file:
2145.             rtext=file.read()
2146.         rtext=Virayesh_Per(rtext)
2147.         jnames_scores_Per=rtext.split("\r\n")
2148.         with codecs.open(files_dir+"Per-AuthNames.txt", 'r',encoding='utf-8') as file:
2149.             rtext=file.read()
2150.         rtext=Virayesh_Per(rtext)
2151.         names_Per=rtext.split("\r\n")
2152.         with codecs.open(files_dir+"\\Per-TitWords.txt", 'r',encoding='utf-8') as file:
2153.             rtext=file.read()
2154.         rtext=Virayesh_Per(rtext)
2155.         tw_scores_Per=rtext.split("\r\n")
2156.         with codecs.open(files_dir+"\\Per-cn.txt", 'r',encoding='utf-8') as file:
2157.             rtext=file.read()
2158.         rtext=Virayesh_Per(rtext)
2159.         cnames_scores_Per=rtext.split("\r\n")
2160.         with codecs.open(files_dir+"\\Eng-jn.txt", 'r',encoding='utf-8') as file:
2161.             rtext=file.read()
2162.         rtext=Virayesh_Eng(rtext)
2163.         jnames_Eng=rtext.split("\r\n")
2164.         with codecs.open(files_dir+"\\Eng-cn.txt", 'r',encoding='utf-8') as file:
2165.             rtext=file.read()
2166.         rtext=Virayesh_Eng(rtext)
2167.         cnames_Eng=rtext.split("\r\n")
2168.         with codecs.open(files_dir+"\\Eng-pn.txt", 'r',encoding='utf-8') as file:
2169.             rtext=file.read()
2170.         rtext=Virayesh_Eng(rtext)
2171.         pnames_Eng=rtext.split("\r\n")
2172.
2173.
2174.         def prepare():
2175.             with codecs.open(files_dir+"\\Citation-Export-Outputs.txt", 'r',encoding='utf-8') as file:
2176.                 taged_text=file.read()
2177.             taged_pars2=taged_text.split("\r\n")
2178.             taged_pars=[]
2179.             for taged_par in taged_pars2:
2180.                 if len(list(taged_par))>30 and len(taged_par.split(" "))>5:
2181.                     taged_pars.append(taged_par)
2182.             fvectors=[]
2183.             for i in list(range(len(taged_pars))):
2184.                 taged_par=taged_pars[i]
2185.                 par_tag=taged_par[:3]
2186.                 par_text=taged_par
2187.                 fvector=Extract_features_Both(par_text)
2188.                 bool_tag=0
2189.                 if par_tag=="111":
2190.                     bool_tag=1
2191.                 fvector.append(bool_tag)
2192.                 fvectors.append(fvector)
2193.             df1=pd.DataFrame(fvectors)
2194.             x1=df1.drop(df1.columns[[len(fvectors[0])-1]],axis=1)
2195.             y1=df1[df1.columns[len(fvectors[0])-1]]
2196.             svclassifier_ext_refs.fit(x1, y1)
2197.
2198.             with codecs.open(files_dir+"\\Eng-Taged-Refs-Type-Train.txt", 'r',encoding='utf-8') as file:
2199.                 taged_text=file.read()
2200.             taged_pars2=taged_text.split("\r\n")
2201.             taged_pars=[]
2202.             for taged_par in taged_pars2:
2203.                 if len(list(taged_par))>30 and len(taged_par.split(" "))>5:
2204.                     taged_pars.append(taged_par)
```

```
2205.            fvectors=[]
2206.            for i in list(range(len(taged_pars))):
2207.                taged_par=taged_pars[i]
2208.                par_tag=taged_par[:3]
2209.                par_text=taged_par[3:]
2210.                fvector=Extract_features_type_Eng(par_text)
2211.                tag=0
2212.                if par_tag=="JJJ":
2213.                    tag=1
2214.                elif par_tag=="CCC":
2215.                    tag=2
2216.                elif par_tag=="BBB":
2217.                    tag=3
2218.                elif par_tag=="TTT":
2219.                    tag=4
2220.                fvector.append(tag)
2221.                fvectors.append(fvector)
2222.            df1=pd.DataFrame(fvectors)
2223.            x1=df1.drop(df1.columns[[len(fvectors[0])-1]],axis=1)
2224.            y1=df1[df1.columns[len(fvectors[0])-1]]
2225.            svclassifier_Type_Eng.fit(x1, y1)
2226.
2227.
2228.            with codecs.open(files_dir+"\\Per-Taged-Refs-Type-Train.txt", 'r',encoding='utf-8') as file:
2229.                taged_text=file.read()
2230.            taged_pars2=taged_text.split("\r\n")
2231.            taged_pars=[]
2232.            for taged_par in taged_pars2:
2233.                if len(list(taged_par))>30 and len(taged_par.split(" "))>5:
2234.                    taged_pars.append(taged_par)
2235.            fvectors=[]
2236.            for i in list(range(len(taged_pars))):
2237.                taged_par=taged_pars[i]
2238.                par_tag=taged_par[:3]
2239.                par_text=taged_par[3:]
2240.                fvector=Extract_features_type_Per(par_text)
2241.                tag=0
2242.                if par_tag=="JJJ":
2243.                    tag=1
2244.                elif par_tag=="CCC":
2245.                    tag=2
2246.                elif par_tag=="BBB":
2247.                    tag=3
2248.                elif par_tag=="TTT":
2249.                    tag=4
2250.                fvector.append(tag)
2251.                fvectors.append(fvector)
2252.            df1=pd.DataFrame(fvectors)
2253.            x1=df1.drop(df1.columns[[len(fvectors[0])-1]],axis=1)
2254.            y1=df1[df1.columns[len(fvectors[0])-1]]
2255.            svclassifier_Type_Per.fit(x1, y1)
2256.
2257.            wb=load_workbook(files_dir+'\Eng-Final-1000-4x250.xlsx')
2258.            mylist=xlsx_to_list(wb)
2259.            myflist=add_neighbours_features_Eng(mylist)
2260.            myfflist=strfeatures_to_intfeatures(myflist)
2261.            df1=pd.DataFrame(myfflist)
2262.            x1=df1.drop(df1.columns[[4,48]],axis=1)
2263.            y1=df1[df1.columns[48]]
2264.            svclassifier_Parse_Eng.fit(x1, y1)
2265.
```

```
2266.             wb=load_workbook(files_dir+'\Per-Final-1000-4x250.xlsx')
2267.             mylist=xlsx_to_list(wb)
2268.             myflist=add_neighbours_features_Per(mylist)
2269.             myfflist=strfeatures_to_intfeatures(myflist)
2270.             df1=pd.DataFrame(myfflist)
2271.             x1=df1.drop(df1.columns[[len(mylist[0])+2]],axis=1)
2272.             y1=df1[df1.columns[len(mylist[0])+2]]
2273.             svclassifier_Parse_Per.fit(x1, y1)
2274.
2275.             wb=load_workbook('C:/E/Fin-Name-Labeling.xlsx')
2276.             mylist=xlsx_to_list(wb)
2277.             myflist=add_neighbours_features_NameLabeling(mylist)
2278.             df1=pd.DataFrame(myflist)
2279.             x1=df1.drop(df1.columns[[0,22,23]],axis=1)
2280.             y1=df1[df1.columns[22]]
2281.             svclassifier_NameLabeling.fit(x1, y1)
2282.
2283.
2284.         class Reference:
2285.
2286.             def __init__(self, ref_lang, ref_type, ref_authors, ref_title, ref_year, ref_venue, ref_volume, ref_pa
    ges,ref_address,ref_seri,ref_month,ref_text):
2287.                 self.lang=ref_lang
2288.                 self.type=ref_type
2289.                 self.authors = ref_authors
2290.                 self.title=ref_title
2291.                 self.year=ref_year
2292.                 self.venue=ref_venue
2293.                 self.volume=ref_volume
2294.                 self.pages=ref_pages
2295.                 self.address=ref_address
2296.                 self.seri=ref_seri
2297.                 self.month=ref_month
2298.                 self.text=ref_text
2299.
2300.
2301.         def lang(text):#### taeen zaban matn voodi (farsi ya englisi)
2302.             num_per=0
2303.             num_eng=0
2304.             for c in list(text):
2305.                 if c in peralph:
2306.                     num_per+=1
2307.                 elif c in engalph+ENGALPH:
2308.                     num_eng+=1
2309.             if num_per>=num_eng:
2310.                 return "Persian"
2311.             else:
2312.                 return "English"
2313.         def is_a_author_match(auth11,auth22):
2314.             auth1=auth11.split("@")
2315.             auth2=auth22.split("@")
2316.             try:
2317.                 if auth1[0]==auth2[0]:
2318.                     if auth1[1][:1]==auth2[1][:1]:
2319.                         return True
2320.             except:
2321.                 return False
2322.             return False
2323.
2324.         def find_ind_matching_author(new_auth):
2325.             for auth in authors_table:
```

```
2326.                 if is_a_author_match(auth[1],new_auth):
2327.                     return auth[0]
2328.             return -1
2329.
2330.         def n_grams(n,text):
2331.             text=text.replace("&","and")
2332.             text=text.replace(" ","")
2333.             text=text.replace(" ","")
2334.             n_gs=[]
2335.             if len(text)<n:
2336.                 n_gs.append(text)
2337.             else:
2338.                 for i in list(range(len(text)-n+1)):
2339.                     n_gs.append(text[i:i+n])
2340.             return n_gs
2341.         def jaccard_similarity(list1, list2):
2342.             intersection = len(list(set(list1).intersection(list2)))
2343.             union = (len(list1) + len(list2)) - intersection
2344.             return float(intersection) / union
2345.         def overlap_similarity(list1,list2):
2346.             intersection = len(list(set(list1).intersection(list2)))
2347.             minn = min(len(list1) , len(list2))
2348.             return float(intersection) / minn
2349.         def levenshtein_similarity(seq1, seq2):
2350.             size_x = len(seq1) + 1
2351.             size_y = len(seq2) + 1
2352.             matrix = np.zeros ((size_x, size_y))
2353.             for x in range(size_x):
2354.                 matrix [x, 0] = x
2355.             for y in range(size_y):
2356.                 matrix [0, y] = y
2357.
2358.             for x in range(1, size_x):
2359.                 for y in range(1, size_y):
2360.                     if seq1[x-1] == seq2[y-1]:
2361.                         matrix [x,y] = min(
2362.                             matrix[x-1, y] + 1,
2363.                             matrix[x-1, y-1],
2364.                             matrix[x, y-1] + 1
2365.                         )
2366.                     else:
2367.                         matrix [x,y] = min(
2368.                             matrix[x-1,y] + 1,
2369.                             matrix[x-1,y-1] + 1,
2370.                             matrix[x,y-1] + 1
2371.                         )
2372.             return float(1)-(matrix[size_x - 1, size_y - 1])/max(size_x,size_y)
2373.         def fittest_venue_match(new_venue):
2374.             possible_matches=[]
2375.             for ven in venues_table:
2376.                 out=is_a_venue_match(ven[1],new_venue)
2377.                 if out[1]==-2:
2378.                     return -2
2379.                 if out[1]>0:
2380.                     possible_matches.append([ven[0],out[1]])
2381.             if len(possible_matches)<1:
2382.                 return -1
2383.             fittest=possible_matches[0][0]
2384.             sim=possible_matches[0][1]
2385.             for p in possible_matches:
2386.                 if p[1]>sim:
```

```
2387.                 fittest=p[0]
2388.                 sim=p[1]
2389.             return fittest
2390.
2391.         def is_a_venue_match(ven11,ven22):
2392.             stopws=[" & "," and "," a "," an "," the "]
2393.             output=[]
2394.             n=3
2395.             t1=0.7
2396.             t11=0.6
2397.             t2=0.8
2398.             t12=0.7
2399.             t3=0.9
2400.             ven1=ven11.lower()
2401.             ven2=ven22.lower()
2402.             if lang(ven1)=="Persian":
2403.                 ven1=Virayesh_Per(ven1)
2404.             else:
2405.                 ven1=Virayesh_Eng(ven1)
2406.             if lang(ven2)=="Persian":
2407.                 ven2=Virayesh_Per(ven2)
2408.             else:
2409.                 ven2=Virayesh_Eng(ven2)
2410.             for sw in stopws:
2411.                 ven1=ven1.replace(sw," ")
2412.                 ven2=ven2.replace(sw," ")
2413.             ven1=ven1.replace("  "," ")
2414.             ven2=ven2.replace("  "," ")
2415.             while ven1[:1]==" ":
2416.                 ven1=ven1[1:]
2417.             while ven2[-1:]==" ":
2418.                 ven2=ven2[:-1]
2419.             output.append(ven1)
2420.             lang_1=lang(ven1)
2421.             lang_2=lang(ven2)
2422.             if lang_1!=lang_2:
2423.                 output.append(0)
2424.                 return output
2425.             v_1=n_grams(n,ven1)
2426.             v_2=n_grams(n,ven2)
2427.             ven_111=ven1
2428.             ven_222=ven2
2429.             for cw in common_words_in_venues:
2430.                 ven_111=ven_111.replace(cw,"")
2431.                 ven_222=ven_222.replace(cw,"")
2432.             if len(ven_222.replace(" ",""))<2:
2433.                 output.append(-2)
2434.                 return output
2435.             if lang_1=="Persian":
2436.                 if jaccard_similarity(v_1,v_2)>t1:
2437.                     if levenshtein_similarity(ven1,ven2)>t11:
2438.                         output.append(levenshtein_similarity(ven1,ven2))
2439.                 elif overlap_similarity(n_grams(3,ven_111),n_grams(3,ven_222))>t3 and math.fabs(len(n_gram
      s(3,ven_111))-len(n_grams(3,ven_222)))<10:
2440.                     output.append(overlap_similarity(v_1,v_2))
2441.
2442.             else:
2443.                 if jaccard_similarity(v_1,v_2)>t2:
2444.                     if levenshtein_similarity(ven1,ven2)>t12:
2445.                         output.append(levenshtein_similarity(ven1,ven2))
```

```
2446.                 elif overlap_similarity(n_grams(3,ven_111),n_grams(3,ven_222))>t3 and math.fabs(len(n_gram
      s(3,ven_111))-len(n_grams(3,ven_222)))<10:
2447.                     output.append(overlap_similarity(v_1,v_2))
2448.             if len(output)==1:
2449.                 output.append(0)
2450.             return output
2451.
2452.         def is_a_match_title(tit11,tit22):
2453.             n=3
2454.             t1=0.7
2455.             t11=0.5
2456.             tit1=tit11.lower()
2457.             tit2=tit22.lower()
2458.             tit1=Virayesh_Both(tit1)
2459.             tit2=Virayesh_Both(tit2)
2460.             lang_1=lang(tit1)
2461.             lang_2=lang(tit2)
2462.             if lang_1!=lang_2:
2463.                 return 0
2464.             t_1=n_grams(n,tit1)
2465.             t_2=n_grams(n,tit2)
2466.             if jaccard_similarity(t_1,t_2)>t1:
2467.                 if levenshtein_similarity(tit1,tit2)>t11:
2468.                     return levenshtein_similarity(tit1,tit2)
2469.             return 0
2470.
2471.         def add_a_new_document(doc):
2472.             ind_of_authors=[]
2473.             for auth in doc.authors:
2474.                 ind=find_ind_matching_author(auth)
2475.                 if ind==-1:
2476.                     ind=len(authors_table)
2477.                     authors_table.append([len(authors_table),auth])
2478.                 ind_of_authors.append(ind)
2479.             ind_of_venue=fittest_venue_match(doc.venue)
2480.             if ind_of_venue==-1:
2481.                 ind_of_venue=len(venues_table)
2482.                 venues_table.append([ind_of_venue,doc.venue])
2483.             ind_of_doc=len(documents_table)
2484.             documents_table.append([ind_of_doc,doc.lang,doc.type,ind_of_authors,doc.title,doc.year,ind_of_
      venue,doc.volume,doc.pages,doc.seri,doc.month,doc.address])
2485.             list_of_refs=ext_refs(doc)
2486.             refs_texts=list_of_refs.split("\r\n")
2487.             k=0
2488.             for ref_t in refs_texts:
2489.                 flag=False
2490.                 ref_text=ref_t
2491.                 if ref_text[:1]=="[":
2492.                     while ref_text[:1]=="]":
2493.                         ref_text=ref_text[1:]
2494.                 if ref_text[:1]=="(":
2495.                     while ref_text[:1]==")":
2496.                         ref_text=ref_text[1:]
2497.                 k+=1
2498.                 ind_of_authors=[]
2499.                 authors_existance=[]
2500.                 lang_=lang(ref_text)
2501.                 if lang_=="Persian":
2502.                     type_=detect_type_Per(ref_text)
2503.                     ref=Reference("Persian",type_,[],"N/A","N/A","N/A","N/A","N/A","N/A","N/A","N/A",ref_text)
2504.                     parsed_ref=parse_Per(ref)
```

```
2505.                elif lang_=="English":
2506.                    type_=detect_type_Eng(ref_text)
2507.                    ref=Reference("English",type_,[],"N/A","N/A","N/A","N/A","N/A","N/A","N/A","N/A",ref_text)
2508.                    parsed_ref=parse_Eng(ref)
2509.                for auth in parsed_ref.authors:
2510.                    ind=find_ind_matching_author(auth)
2511.                    ind_of_authors.append(ind)
2512.
2513.                ind_of_venue=-2
2514.                if len(list(parsed_ref.venue))>5:
2515.                    ind_of_venue=fittest_venue_match(parsed_ref.venue)
2516.                if ind_of_venue==-1:
2517.                    ind_of_venue=len(venues_table)
2518.                    venues_table.append([ind_of_venue,parsed_ref.venue])
2519.                ind_of_ref=-1
2520.                if len(ind_of_authors)>0:
2521.                    ind_of_ref=fittest_ref_match(parsed_ref,ind_of_authors,ind_of_venue)
2522.                if ind_of_ref==-1:
2523.                    new_ind_of_authors=[]
2524.                    for i in list(range(len(parsed_ref.authors))):
2525.                        if ind_of_authors[i]==-1:
2526.                            new_ind_of_authors.append(len(authors_table))
2527.                            authors_table.append([len(authors_table),parsed_ref.authors[i]])
2528.                        else:
2529.                            new_ind_of_authors.append(ind_of_authors[i])
2530.                    ind_of_ref=len(documents_table)
2531.                    documents_table.append([ind_of_ref,lang_,type_,new_ind_of_authors,parsed_ref.title,parsed
      _ref.year,ind_of_venue,parsed_ref.volume,
2532.                                            parsed_ref.pages,parsed_ref.address,parsed_ref.seri,parsed_ref.month])
2533.
2534.                citations_table.append([ind_of_ref,ind_of_doc])
2535.
2536.            def fittest_ref_match(new_ref,ind_of_authors,ind_of_venue):
2537.                possible_matches=[]
2538.                for doc in documents_table:
2539.                    if new_ref.lang==doc[1] and new_ref.type==doc[2]:
2540.                        if len(doc[3])>0:
2541.                            if doc[3][0]==ind_of_authors[0]:
2542.                                tit_match=is_a_match_title(doc[4],new_ref.title)
2543.                                if tit_match>0:
2544.                                    possible_matches.append([tit_match,doc[0]])
2545.                if len(possible_matches)>=1:
2546.                    out_ind=possible_matches[0][1]
2547.                    sim=possible_matches[0][0]
2548.                    for p in possible_matches:
2549.                        if p[0]>sim:
2550.                            out_ind=p[1]
2551.                            sim=p[0]
2552.                    return out_ind
2553.                else:
2554.                    for doc in documents_table:
2555.                        if new_ref.lang==doc[1] and new_ref.type==doc[2]:
2556.                            if new_ref.year==doc[5] and ind_of_venue==doc[6]:
2557.                                if len(set(ind_of_authors).intersection(set(doc[3])))>0:
2558.                                    if jaccard_similarity(n_grams(3,doc[4]),n_grams(3,new_ref.title))>0.6:
2559.                                        return doc[0]
2560.                                elif new_ref.volume==doc[7] and new_ref.pages==doc[8]:
2561.                                    if jaccard_similarity(n_grams(3,doc[4]),n_grams(3,new_ref.title))>0.6:
2562.                                        return doc[0]
2563.
2564.                    return -1
```

```
2565.
2566.           prepare()
2567.           for i in list(range(1,101)):
2568.               start = time.time()
2569.               text = docx2txt.process(files_dir+"\\Inputs\\"+str(i)+".docx")
2570.               doc=Document("Persian","Thesis",["N. Pakniat"],"Title","1395","venue","1","20-
    30","address","1","Jun",text)
2571.               add_a_new_document(doc)
2572.               elapsed_time = (time.time() - start)
2573.               print(elapsed_time)
2574.               print("----------------------")
2575.
2576.           tmp=""
2577.           for doc in documents_table:
2578.               for c in doc:
2579.                   tmp=tmp+str(c)+"\t"
2580.               tmp=tmp[:-1]
2581.               tmp=tmp+"\r\n"
2582.           with codecs.open(files_dir+"\\Outputs\\Docs-Table.txt", 'w',encoding='utf-8') as file:
2583.               file.write(tmp)
2584.           tmp=""
2585.           for auth in authors_table:
2586.               for a in auth:
2587.                   tmp=tmp+str(a)+"\t"
2588.               tmp=tmp[:-1]
2589.               tmp=tmp+"\r\n"
2590.           with codecs.open(files_dir+"\\Outputs\\Auths-Table.txt", 'w',encoding='utf-8') as file:
2591.               file.write(tmp)
2592.           tmp=""
2593.           for citation in citations_table:
2594.               for c in citation:
2595.                   tmp=tmp+str(c)+"\t"
2596.               tmp=tmp[:-1]
2597.               tmp=tmp+"\r\n"
2598.           with codecs.open(files_dir+"\\Outputs\\Citations-Table.txt", 'w',encoding='utf-8') as file:
2599.               file.write(tmp)
2600.           tmp=""
2601.           for ven in venues_table:
2602.               for v in ven:
2603.                   tmp=tmp+str(v)+"\t"
2604.               tmp=tmp[:-1]
2605.               tmp=tmp+"\r\n"
2606.           with codecs.open(files_dir+"\\Outputs\\Venues-Table.txt", 'w',encoding='utf-8') as file:
2607.               file.write(tmp)
```

پیوست ۲: مستندات فنی ماژول نرم‌افزاری تهیه شده

در این بخش، مستندات فنی متناظر با برنامه نهایی تهیه و ارائه شده ارائه می‌شود.

راهنمای کد نرم‌افزار:

زبان برنامه‌نویسی و بسته‌های استفاده شده

در این بخش بسته[۱]ها و زبان برنامه‌نویسی استفاده شده در این پروژه معرفی می‌شوند.

زبان برنامه‌نویسی استفاده شده برای تهیه برنامه ارائه شده #C است که از فریم‌ورک Net Core 2.2. استفاده می‌نماید. برنامه ارائه شده یک Console application است که با افزودن dll آن، می‌توان از آن در پروژه‌های دیگر استفاده کرد. برای ساخت پایگاه داده برنامه از Entity Framework Code-First استفاده شده است. به‌کارگیری روش‌های یادگیری ماشینی در این برنامه با استفاده از بسته Encog صورت گرفته که یکی از بسته‌های شناخته شده در زمینه هوش مصنوعی برای زبان‌های جاوا و #C است.

همچنین از بسته Microsoft تحت عنوان DocumentFormat.OpenXml برای خواندن اسناد doc و docx استفاده شده است.

کلاس‌های موجود در برنامه تهیه شده

در این بخش به معرفی کلاس‌های موجود در برنامه تهیه شده می‌پردازیم.

کلاس Fvectors: این کلاس یک ساختار داده برای ویژگی‌ها و برچسب‌های استخراج شده در روش SVM است.

کلاس Mydata: این کلاس ساختار داده به دست آمده را برای استفاده در روش SVM آماده می‌نماید.

کلاس MyDocument: این کلاس یک ساختار داده برای اسناد است که اطلاعات هر سند را نگهداری می‌نماید.

---

[۱] Package

کلاس MyReference: این کلاس یک ساختار داده برای مراجع است که اطلاعات هر مرجع در آن نگهداری می‌شود.

کلاس Article: این کلاس برای ایجاد و ارتباط با پایگاه داده می‌باشد که تنها اطلاعات مقاله مانند عنوان، صفحه، نوع و ... در آن درج می‌شود.

کلاس Citation: ارجاع به مقالات به کمک این کلاس در جدول ارجاعات ذخیره می‌شود.

کلاس Contribution: این کلاس واسط بین اسناد و نویسندگان است.

کلاس Organization: نشریه، همایش، انتشارت یا دانشگاه منتشر کننده اسناد توسط این کلاس در پایگاه داده ذخیره می‌شود.

کلاس Partnership: این کلاس واسط جدول اسناد و واحدهای منتشرکننده آن‌هاست.

کلاس Researcher: این کلاس برای ذخیره اطلاعات پژوهشگران ایجاد شده است.

کلاس MainDoc: این کلاس مربوط به فرادادهها و فایل‌های پارساهای ورودی است. با استفاده از این کلاس، فرادادهها و فایل‌های پارساها مستقیما از پایگاه داده خوانده می‌شود. در نتیجه وجود این کلاس، نیازی به تغییر در برنامه برای افزودن پارساهای جدید نیست.

توابع

در این بخش به توصیف توابع به کاررفته در برنامه تهیه شده پرداخته شده است.

تابع ProcessParagraph: از این تابع برای اتصال پاراگراف‌های خوانده شده به یکدیگر استفاده می‌شود.

تابع sep_auths: این تابع یک رشته ورودی شامل نام چندین پژوهشگر را به عنوان ورودی دریافت کرده و پس از تفکیک، لیست پژوهشگران را خروجی می‌دهد.

تابع is_f_name_indicator: این تابع برای بررسی وجود یک واحد در قالب مخفف نام یک نویسنده ایجاد شده است.

تابع is_contain_nonalphabetic: این تابع وجود کاراکترهای غیرالفبایی در کاراکترهای متناظر با واحد ورودی را بررسی می‌کند.

تابع is_contain_num_alph: این تابع وجود همزمان حداقل یک کاراکتر الفبایی و یک کاراکتر عددی در مجموعه کاراکترهای متناظر با واحد ورودی را بررسی می‌کند.

تابع is_only_contain_num_alph: این تابع وجود کاراکتری غیر ازکاراکترهای الفبایی و عددی در واحد ورودی را بررسی می‌کند.

تابع is_et_al_Per: این تابع وجود واحد ورودی در مجموعه {دیگران، همکاران} را بررسی می‌کند.

تابع is_j_ind_Per: این تابع وجود واحد ورودی در مجموعه نشان‌گرهای نام نشریه در زبان فارسی را بررسی می‌کند.

تابع is_c_ind_Per: این تابع وجود واحد در مجموعه نشان‌گرهای نام همایش در زبان فارسی را بررسی می‌کند.

تابع is_author_name_Per: این تابع وجود واحد ورودی در لیست اسامی پژوهشگران به زبان فارسی را بررسی می‌کند.

تابع j_name_score_Per: این تابع، با توجه به مجموعه‌ای از پیش جمع‌آوری شده از نام‌های نشریات فارسی‌زبان، امتیازی مبنی بر احتمال نام نشریه بودن واحد ورودی به آن اختصاص می‌دهد.

تابع c_name_score_Per: این تابع، با توجه به مجموعه‌ای از پیش جمع‌آوری شده از نام‌های همایش‌های فارسی‌زبان، امتیازی مبنی بر احتمال نام همایش بودن واحد ورودی به آن اختصاص می‌دهد.

تابع title_name_score_Per: این تابع، با توجه به مجموعه‌ای از پیش جمع‌آوری شده از عناوین اسناد فارسی‌زبان، امتیازی مبنی بر احتمال اینکه واحد ورودی بخشی از عنوان یک سند باشد، به آن اختصاص می‌دهد.

تابع is_in_pn_Eng: این تابع وجود واحد ورودی در لیست واژگان نام‌های انتشارات شناخته شده در زبان انگلیسی را بررسی می‌کند.

تابع is_in_jn_Eng: این تابع وجود واحد در لیست واژگان اسامی نشریات انگلیسی‌زبان شناخته شده را بررسی می‌کند.

تابع is_in_cn_Eng: این تابع وجود واحد در لیست واژگان اسامی همایش‌های انگلیسی‌زبان شناخته شده را بررسی می‌کند.

تابع is_containing_Eng_alphabet: این تابع وجود حداقل یک حرف انگلیسی در واحد ورودی را بررسی می‌کند.

تابع replace_sep_with_sepspace: این تابع، به جستجوی علائم نقطه‌گذاری در رشته ورودی پرداخته و در صورت عدم وجود کاراکتر فاصله پس از این علائم در رشته ورودی، با اضافه کردن فاصله رشته ورودی را ویرایش می‌نماید.

تابع Virayesh_Both: این تابع ویرایش‌های موردنیاز برای فرایندهای مستقل از زبان را روی متون انجام می‌دهد.

تابع Virayesh_Per: این تابع ویرایش‌های موردنیاز را روی متون فارسی زبان انجام می‌دهد.

تابع Virayesh_Eng: این تابع ویرایش‌های موردنیاز را روی متون انگلیسی زبان انجام می‌دهد.

تابع is_a_written_number1: این تابع به بررسی وجود واحد ورودی درمجموعه $\{1^{st}, 2^{nd}, 3^{rd}, 4^{th}, \ldots\}$ می‌پردازد.

تابع is_a_written_number2: این تابع به بررسی وجود واحد ورودی در مجموعه {first, second, third, … ,forth} می‌پردازد.

تابع is_a_written_number: این تابع به بررسی وجود واحد ورودی در مجموعه {یک، دو، سه، ... اولین، دومین و ...} می‌پردازد.

تابع is_mon_seas_name_Per: این تابع وجود واحد ورودی در مجموعه نام‌های متناظر با ماه‌ها و فصل‌های سال در زبان فارسی را بررسی می‌کند.

تابع is_mon_seas_name_Eng: این تابع وجود واحد ورودی در مجموعه نام‌های متناظر با ماه‌ها و فصل‌های سال در زبان انگلیسی را بررسی می‌کند.

تابع contains_volume_ind_Both: این تابع وجود واحد ورودی در لیست مشخص‌کننده‌های دوره چاپ نشریات را بررسی می‌کند.

تابع contains_num_ind_Both: این تابع وجود واحد ورودی در لیست مشخص‌کننده‌های شماره چاپ نشریات را بررسی می‌کند.

تابع contains_series_ind_Both: این تابع وجود واحد ورودی در لیست مشخص‌کننده‌های شماره چاپ کتاب‌ها را بررسی می‌کند.

تابع contains_parts_ind_Per: این تابع وجود واحد ورودی در لیست مشخص‌کننده‌های شماره جلد کتاب‌ها را بررسی می‌کند.

تابع contains_page_ind_Both: این تابع وجود واحد ورودی در لیست مشخص‌کننده‌های شماره صفحه چاپ را بررسی می‌کند.

تابع is_page_number_Both: این تابع وجود واحد ورودی در قالب شماره صفحه چاپ را بررسی می‌کند.

تابع sep_text_from_number: این تابع به جداسازی اعداد از مشخص‌کننده‌های دوره، شماره، صفحه، جلد و ... می‌پردازد.

تابع dash_editor: این تابع فواصل موجود بین اعداد و کاراکتر "-" را در صورت وجود حذف می‌نماید.

تابع is_number: این تابع عدد بودن واحد ورودی را بررسی می‌کند.

تابع hasNumbers: این تابع وجود حداقل یک کاراکتر عددی در لیست کاراکترهای واحد ورودی را بررسی می‌کند.

تابع is_year_number: این تابع نشانگر سال بودن واحد ورودی را بررسی می‌کند.

تابع contain_l_name_indicator: این تابع ختم واحد ورودی به یکی از پسوندهای رایج نام‌خانوادگی در زبان‌های فارسی (زاده، پور، فر، نیا و ...) را بررسی می‌کند.

تابع special_year_editting: این تابع به ویرایش واحدهای نماشگر سال که به صورت عدد سال همراه با حرف اول نوع سال هستند (مانند ۱۳۶۵ش)، می‌پردازد.

تابع Convertnumbs: این تابع یکسان‌نویسی اعداد را انجام می‌دهد.

تابع replace_sep_with_space: این تابع علائم نقطه‌گذاری موجود در رشته ورودی را به فاصله جایگزین می‌کند.

تابع is_ended_with_seperator: این تابع ختم واحد ورودی به علائم نقطه‌گذاری را بررسی می‌کند.

تابع Extract_features_Both: این تابع، با ورودی یک پاراگراف، بردار ویژگی متناظر با آن را جهت تعیین رشته مرجع بودن آن پاراگراف استخراج می‌نماید.

تابع Extract_features_type_Eng: این تابع، با ورودی یک رشته مرجع به زبان انگلیسی، بردار ویژگی متناظر با آن را جهت تعیین نوع رشته ورودی استخراج می‌کند.

تابع Extract_features_type_Per: این تابع، با ورودی یک رشته مرجع به زبان فارسی، بردار ویژگی متناظر با آن را جهت تعیین نوع رشته ورودی استخراج می‌کند.

تابع Extract_features_Parse_Eng: این تابع، با ورودی یک واحد (واژه) از یک رشته مرجع در زبان انگلیسی، بردار ویژگی متناظر با آن را جهت تعیین برچسب آن واحد استخراج می‌نماید.

تابع Extract_features_Parse_Per: این تابع، با ورودی یک واحد (واژه) از یک رشته مرجع در زبان فارسی، بردار ویژگی متناظر با آن را جهت تعیین برچسب آن واحد استخراج می‌نماید.

تابع Extract_features_NameLabeling: این تابع، با ورودی یک واحد، بردار ویژگی متناظر با آن را جهت تعیین نام یا نام خانوادگی بودن آن واحد استخراج می‌نماید.

تابع ext_refs: این تابع، با ورودی تمام متن یک پارسا، لیست مراجع آن را استخراج می‌کند.

تابع detect_type_Eng: این تابع نوع رشته‌های مرجع به زبان انگلیسی را مشخص می‌کند.

تابع detect_type_Per: این تابع نوع رشته‌های مرجع به زبان فارسی را مشخص می‌کند.

تابع add_neighbours_features_Per: در روش‌های ارائه شده برای تجزیه رشته‌های مرجع، علاوه بر ویژگی‌های متناظر با یک واحد، از برچسب‌های واحدهای قبل از آن و همچنین ویژگی‌های واحد بعد از آن نیز برای تعیین برچسب واحد فعلی استفاده می‌شود. این کار در برنامه ارائه شده برای رشته‌های مرجع به زبان فارسی توسط این تابع صورت می‌گیرد.

تابع add_neighbours_features_Eng: در روش‌های ارائه شده برای تجزیه رشته‌های مرجع، علاوه بر ویژگی‌های متناظر با یک واحد، از برچسب‌های واحدهای قبل از آن و همچنین ویژگی‌های واحد بعد از آن نیز برای تعیین برچسب واحد فعلی استفاده می‌شود. این کار در برنامه ارائه شده برای رشته‌های مرجع به زبان انگلیسی توسط این تابع صورت می‌گیرد.

تابع strfeatures_to_intfeatures: با توجه به وجود برچسب‌های واحد قبلی در بردار ویژگی یک واحد، این تابع برچسب‌ها را به عدد تبدیل می‌کند تا توسط دسته‌بندها قابل استفاده باشند.

تابع parse_Per: این تابع، یک رشته مرجع به زبان فارسی را به عنوان ورودی دریافت کرده و پس از تجزیه، مولفه‌های سازنده آن را خروجی می‌دهد.

تابع parse_Eng: این تابع، یک رشته مرجع به زبان انگلیسی را به عنوان ورودی دریافت کرده و پس از تجزیه، مولفه‌های سازنده آن را خروجی می‌دهد.

تابع Prepare: آموزش دسته‌بندهای مورد استفاده در این پژوهش توسط این تابع صورت می‌گیرد.

تابع Lang: این تابع زبان متن ورودی را تعیین می‌کند.

تابع is_a_author_match: این تابع دو نام را به عنوان ورودی دریافت کرده و تطابق آن‌ها را بررسی می‌کند.

تابع find_ind_matching_author: این تابع، با ورودی نام یک پژوهشگر، اندیس متناظر با آن را از جدول اسناد بازمی‌گرداند. لازم به ذکر است که در صورتی که نام موردنظر در جدول نام‌ها موجود نباشد، این تابع ابتدا آن را به جدول اضافه می‌کند.

تابع n_grams: این تابع مجموعه n-تایی‌ها متناظر با یک رشته ورودی را خروجی می‌دهد.

تابع jaccard_similarity: این تابع دو مجموعه را به عنوان ورودی دریافت کرده و میزان شباهت آن‌ها را با توجه به معیار ژاکارد خروجی می‌دهد.

تابع overlap_similarity: این تابع دو مجموعه را به عنوان ورودی دریافت کرده و میزان شباهت آن‌ها را با توجه به ضریب همپوشانی خروجی می‌دهد.

تابع levenshtein_similarity: این تابع دو رشته را به عنوان ورودی دریافت کرده و میزان شباهت آن‌ها را با توجه به معیار لون‌اشتاین خروجی می‌دهد .

تابع fittest_venue_match: این تابع، نام یک نشریه، همایش، انتشارات یا دانشگاه را به عنوان ورودی دریافت کرده و در صورتی که میزان مطابقت با برخی از نام‌های موجود در جدول محل‌ها از مقداری آستانه‌ای بیشتر باشد، مطابق‌ترین را خروجی می‌دهد.

تابع is_a_match_title: این تابع به بررسی تطابق دو عنوان می‌پردازد.

تابع add_a_new_document: فرایند اضافه کردن یک پارسای جدید به پایگاه داده توسط این تابع و با فراخوانی دیگر توابع موجود در این گزارش انجام می‌شود.

تابع fittest_ref_match: این تابع با ورودی مولفه‌های متناظر با یک رشته مرجع، به بررسی تطابق آن با اسناد ذخیره شده در جدول اسناد پرداخته و مطابق‌ترین را، در صورتی که میزان انطباق از مقداری آستانه‌ای بیشتر باشد، بازمی‌گرداند.

تابع ReadExcel: این تابع آدرس یک فایل اکسل را به عنوان ورودی دریافت کرده و پس از خواندن آن فایل، لیستی از رشته‌های موجود در آن را خروجی می‌دهد.

تابع ImportExcel: این تابع آدرس یک فایل اکسل را دریافت کرده و اطلاعات خوانده شده را در قالب لیستی از DataTableها بازمی‌گرداند.

تابع GetTextFromOdt: این تابع آدرس یک فایل open office را گرفته و رشته متناظر با متن فایل موردنظر را به عنوان خروجی بازمی‌گرداند. از این تابع برای خواندن فایل‌هایی با پسوند odt استفاده شده است.

تابع extractfromtex: به کمک این تابع فایل‌های تهیه شده با نرم‌افزار زی‌پرشین خوانده می‌شود. ورودی این تابع، آدرس فایل و خروجی آن متن فایل است. این تابع همچنین مولفه‌های مراجع را نیز استخراج کرده و در پایگاه داده ذخیره می‌کند.

تابع typelatex: این تابع به تشخیص نوع فایل‌های نرم‌افزار زی‌پرشین پرداخته وbib. را از tex. تمایز می‌دهد.

تابع Deletall: از آن‌جا که فایل‌های متناظر با برخی اسناد تهیه شده با نرم‌افزار زی‌پرشین به صورت فشرده شده و در قالب zip موجود هستند، از این تابع برای استخراج فایل‌های فشرده استفاده می‌شود. در این راستا، این تابع یک پوشه برای ذخیره فایل‌های استخراج شده ایجاد کرده و فایل‌ها را در آن ذخیره می‌نماید.

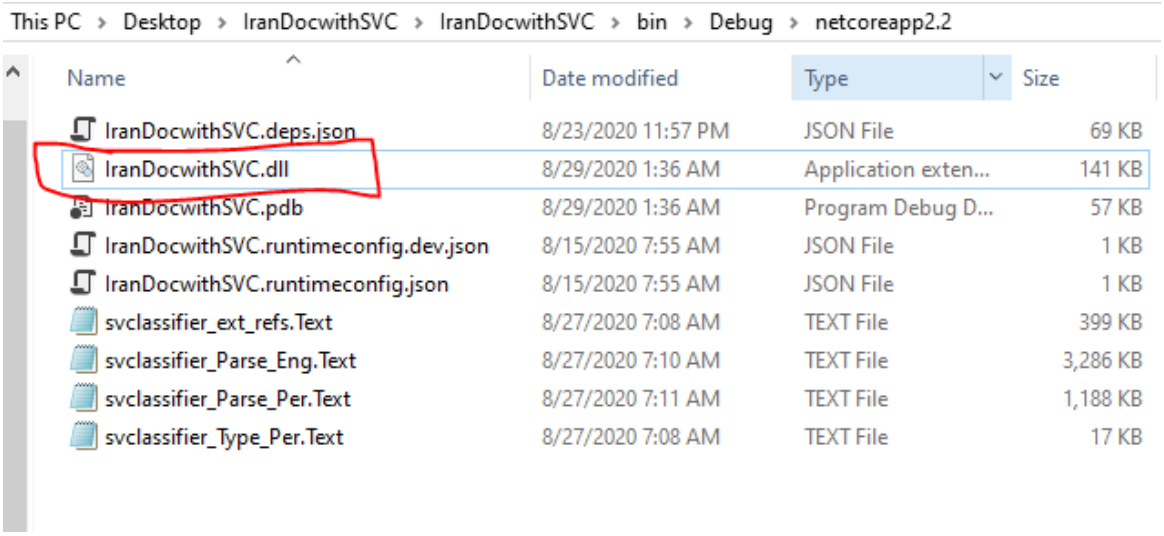
تابع Loadlatex: این تابع به بررسی zip بودن فایل متناظر با یک پارسا می‌پردازد.

تابع OpenWordprocessingDocumentReadonly: از این تابع برای خواندن فایل پارساهای با فرمت doc و docx استفاده شده است.

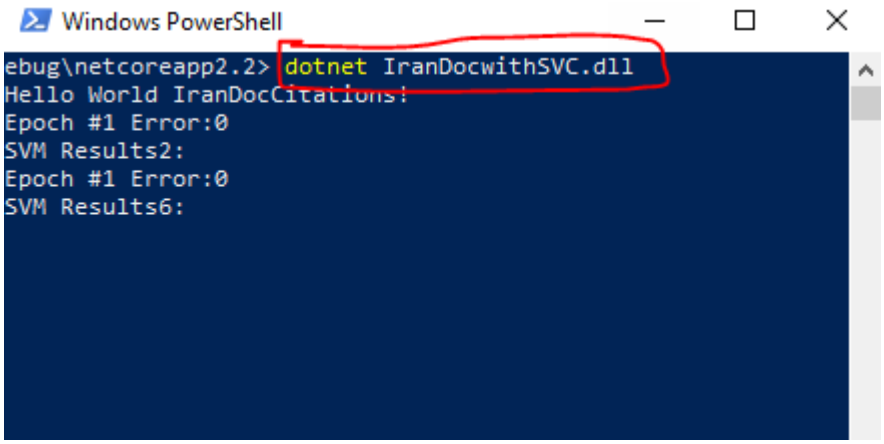
راهنمای نصب و استقرار نرم‌افزار و راهنمای کاربران نرم‌افزار

برای اجرای برنامه تهیه شده در این پژوهش کافی است با استفاده از cmd یا powershell به پوشه مشخص شده در شکل ۵ رفته و دستور زیر، همانطور که در شکل ۶ مشخص شده، فراخوانی شود:

dotnet IranDocwithSVC.dll



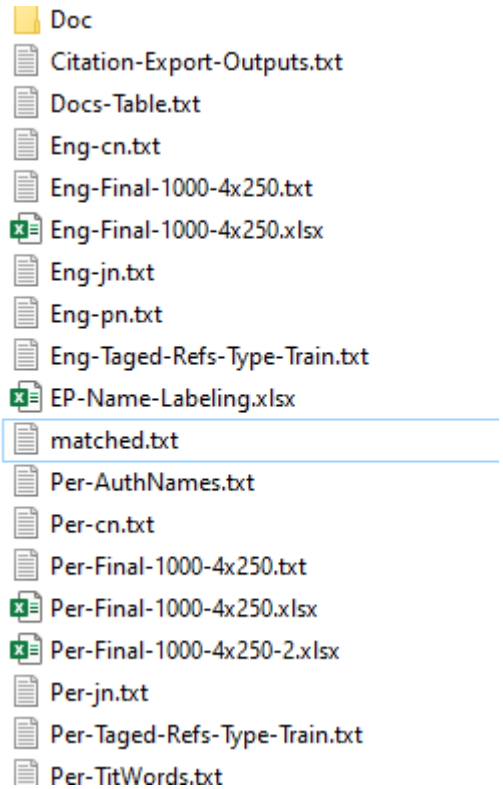
شکل ۵. آدرس ذخیره برنامه ایجاد شده



شکل ٦. دستور راه‌اندازی برنامه

مسیر فایل برنامه

در برنامه ایجاد شده متغییری با نام files_dir وجود دارد. برای استفاده از برنامه، باید فایل‌های مشخص شده در شکل ۷ در آدرس مشخص شده در این متغییر قرار داده شود:



شکل ۷. فایل‌های کمکی موردنیاز برای اجرای برنامه ایجاد شده

مجموعه پارساهایی که قرار است با شبکه تحلیل استنادی اضافه شوند در پوشه Doc مشخص شده در شکل بالا قرار داده می‌شوند. سایر فایل‌ها، فایل‌هایی کمکی هستند که برای مرحله آموزش و استخراج مراجع مورد استفاده قرار می‌گیرند.

مسیر برای فایل‌های مورد نیاز یادگیری ماشینی به صورت پارامتر تعریف شده است که اگر کاربر در کنسول مسیر مورد نظرش را وارد کند آن مسیر در نظر گرفته می شود و اگر مسیری را وارد نکند مسیر پیش زیر در نظر گرفته می شود:

C:\Irandoccitation\E\

مسیر فایل‌های پایان‌نامه جهت ورودی برنامه برای ساخت شبکه تحلیل استنادی به صورت پارامتر تعریف شده است که اگر کاربر در کنسول مسیر مورد نظرش را وارد کند آن مسیر در نظر گرفته می‌شود و اگر مسیری را وارد نکند مسیر پیش‌فرض زیر در نظر گرفته می‌شود.

C:\Irandoccitation\E\Doc\

Abstract

Citations, provided usually at the end of the scientific documents, can be used as a tool to retrieve information of interest. In addition, they can be used as a measure to evaluate the impact or significance of scientific documents and at a higher level, to evaluate individual's performances for promotions and grants. The use of citations for the mentioned purposes requires the construction of the citation networks to enable determination of the relationships between entities such as documents, researchers, journals, universities and research institutes, publications, etc. Traditional (manual) methods for the construction of citation networks on collections of scientific documents are very time consuming and almost impossible for large collections of documents. As a result, we need autonomous methods for construction the citation networks.
Iranian Science Database (GANJ) is one of the most important scientific sources in Persian language. It has been developed by the Iranian Research Institute for Information Science and Technology (IranDoc) and includes most of the completed theses by Iranian researchers. Due to the importance of this database and the lack of methods for autonomous citation network construction in Persian, in this study, we address this problem. In this regard, first, the problem of autonomous citation network construction is divided into some subproblems such as citation extraction, reference type detection, citation parsing and citation matching; and then, considering these problems as classification problems, we use machine learning methods and approximate string matching algorithms to solve them.
It should be noted that due to the frequency of using references to English documents in Persian documents, in addition to Persian-language references, the proposed solutions also consider References written in English language. The used machine learning method in this research is the support vector machine, which is a supervised method. In order to use supervised machine learning methods, there should be available some labeled datasets for the training and the testing of the methods, Due to the lack of such datasets for the Persian language, several such datasets are also created in this study.
At the end, the solutions provided for the mentioned subproblems ofare merged in a software module. The designed module, on input of a collection of scientific documents and their corresponding metadata, constructs the underlying citation network and outputs it in Json format. The obtained precision, recall and F1 measures in overall evaluation of the proposed module are 1, 0.78 and 0.88, respectively, which are quite promising.

Keywords: Citation network, Citation indexing, Citation parsing, Citation matching, Classification, Machine learning, Support vector machine, Levenshtein distance.

Iranian Research Institute for Information Science and Technology (Irandoc)

Research Project

Developing a software module for building the citation network of the existing theses in the Iranian Science Database (GANJ)

By:

Nasrollah Pakniat

Co-authors:

Jalal A. Nasiri

October 2020