
بسم الله الرحمن الرحيم

دانشگاه شاهرود

دانشکده مهندسی برق و رباتیک

گروه رباتیک

تشخیص دستنوشته برخط فارسی با رویکرد تجزیه ای

علیرضا فخری نوش آبادی

استاد راهنما:

دکتر علیرضا احمدی فرد

استاد مشاور:

دکتر حسین خسروی

پایان نامه ارشد جهت اخذ درجه کارشناسی ارشد

بهمن ۹۳

دانشگاه شاهرود

دانشکده :

گروه :

پایان نامه کارشناسی ارشد آقای/ خانم .....

تحت عنوان:

در تاریخ ..... توسط کمیته تخصصی زیر جهت اخذ مدرک کارشناسی ارشد مورد ارزیابی و با درجه ..... مورد پذیرش قرار گرفت.

اساتید راهنما	امضاء	اساتید مشاور	امضاء
نام و نام خانوادگی :		نام و نام خانوادگی :	
نام و نام خانوادگی :		نام و نام خانوادگی :	

اساتید داور	امضاء	نماینده تحصیلات تکمیلی	امضاء
نام و نام خانوادگی :		نام و نام خانوادگی :	
نام و نام خانوادگی :			
نام و نام خانوادگی :			
نام و نام خانوادگی :			

تقدیم به پدر و مادر عزیزم

تشکر و قدردانی

نخستین سپاس و ستایش از آن خداوندی است که بنده کوچکش را در دریای بیکران اندیشه، قطره‌ای ساخت تا وسعت آن را از دریچه اندیشه های ناب آموزگارانی بزرگ به تماشا نشیند. لذا اکنون در سایه سار بنده نوازی هایش پایان نامه حاضر به انجام رسیده است، بر خود لازم می دانم تا مراتب سپاس را از بزرگوارانی به جا آورم که اگر دست یاریگرشان نبود، هرگز این پایان نامه به انجام نمی رسید.

ابتدا از استاد گرانقدرم جناب آقای دکتر احمدی فرد که زحمت راهنمایی این پایان نامه را بر عهده داشتند، بخاطر کمک های بی ردیغ ایشان کمال سپاس را دارم.

از استاد عالی قدرم جناب آقای دکتر خسروی که زحمت مشاوره این پایان نامه را متحمل شدند، صمیمانه تشکر می کنم.

سپاس آخر به مهربان‌ترین همراهان زندگیم، خانواده و دوستان عزیزم، که حضورشان در فضای زندگیم مصداق بی ریا سخاوت بوده است.

تعهد نامه

اینجانب علیرضا فخری نوش آبادی دانشجوی دوره کارشناسی ارشد رشته برق الکترونیک دانشکده برق و رباتیک دانشگاه صنعتی شاهرود نویسنده پایان نامه تشخیص دستنوشته برخط فارسی با رویکرد تجزیه ای تحت راهنمائی دکتر علیرضا احمدی فرد متعهد می شوم :

- تحقیقات در این پایان نامه توسط اینجانب انجام شده است و از صحت و اصالت برخوردار است .
- در استفاده از نتایج پژوهشهای محققان دیگر به مرجع مورد استفاده استناد شده است .
- مطالب مندرج در پایان نامه تاکنون توسط خود یا فرد دیگری برای دریـافـت هیچ نوع مـدرک یا امتیازی در هیچ جا ارائه نشده است .
- کلیه حقوق معنوی این اثر متعلق به دانشگاه صنعتی شاهرود می باشد و مقالات مستخرج با نام « دانشگاه صنعتی شاهرود » و یا « Shahrood University of Technology » به چاپ خواهد رسید .
- حقوق معنوی تمام افرادی که در به دسـت آمدن نتایج اصلی پایان نامه تأثیرگذار بوده اند در مقالات مستخرج از پایان نامه رعایت می گردد.
- در کلیه مراحل انجام این پایان نامه ، در مواردی که از موجود زنده (یا بافتهای آنها) اسـتفاده شـده است ضوابط و اصول اخلاقی رعایت شده است .
- در کلیه مراحل انجام این پایان نامه، در مواردی که به حوزه اطلاعات شخصی افراد دسترسی یافته یا استفاده شده است اصل رازداری ، ضوابط و اصول اخلاق انسانی رعایت شده است .

تاریخ **امضای دانشجو**

مالکیت نتایج و حق نشر

- کلیه حقوق معنوی این اثر و محصولات آن (مقالات مستخرج ، کتاب ، برنامه های رایانه ای ، نرم افزار ها و تجهیزات ساخته شده است) متعلق به دانشگاه صنعتی شاهرود می باشد . این مطلب باید به نحو مقتضی در تولیدات علمی مربوطه ذکر شود .
- استفاده از اطلاعات و نتایج موجود در پایان نامه بدون ذکر مرجع مجاز نمی باشد.

چکیده

در این تحقیق روشی برای تشخیص دستنوشته برخط فارسی بر مبنای شناسایی حروف سازنده زیرکلمات ارائه شده است. ورودی سیستم های تشخیص دستنوشته برخط مجموعه ای از نقاط در صفحه مربوط به دستنوشته است که عموماً توسط وسایل دیجیتال نظیر تبلت ها و تلفن های همراه هوشمند و به طور هم زمان با عمل نوشتن جمع آوری می شوند و به همین دلیل به آن برخط گویند.

در روش پیشنهادی در یک گام ساده سازی داده ورودی به صورت مجموعه ای از خطوط افقی و عمودی با دو بردار جهت و اندازه خطوط که در واقع ویژگی‌های استخراجی از داده هستند بازنمایی می شود. سپس برای تشکیل پایگاه دانش سیستم از بدنه حروف الفبا در مکان‌های مختلف زیرکلمه از بردارهای ویژگی الگوهای مرجع استخراج می‌شوند. در گام شناسایی، لیستی از زیرکلمات با شناسایی حروف سازنده نمونه ورودی بوسیله انطباق الگو با استفاده از یک الگوریتم پیشنهادی با تکنیک برنامه نویسی پویا پیشنهاد می‌شود و درنهایت در گام پس پردازش پیشنهاداتی که در فرهنگ لغت سیستم وجود نداشته باشند و یا از لحاظ تعداد و مکان قرارگیری علامت ها با داده ورودی همخوانی نداشته باشند توسط دو فیلتر حذف می شوند.

درصد تشخیص حروف جدای فارسی با استفاده از ساده سازی و الگوریتم پیشنهادی محاسبه فاصله اصلاح برابر ۹۵/۲ ٪ شده است و درصد بازشناسی برای تشخیص زیرکلمات فارسی برای سه گزینه اول پیشنهادی برابر ۶۳/۲۹٪ بدست آمد.

هرچند درصد تشخیص بدست آمده نسبت به روشهای کلی نگر پایین تر است ولی مزیت روش پیشنهادی در شناسایی حروف سازنده زیرکلمه نسبت به روش‌های کلی نگر در آن است که برای بازشناسی زیرکلمات جدید تنها کافی است که متن این زیرکلمات به فرهنگ لغت سیستم اضافه شوند، درحالی که در روش‌های کلی‌نگر نیاز به نمونه‌های جدید از دستنوشته است.

کلمات کلیدی: تشخیص دستنوشته برخط فارسی، انطباق الگو، فاصله اصلاح

فهرست مطالب

عنوان **صفحه**

فهرست اشکال

عنوان **صفحه**

فهرست جداول

عنوان **صفحه**

علائم و اختصارات

DP	Dynamic Programming
DTW	Dynamic time warping
FEM	Fuzzy Elastic Machine
HMM	Hidden Markov Model
NN	Neural Network
OHRS	Online handwriting recognition system
OCR	Optical Character recognition
PDA	Personal Digital Assistant

فصل ۱

مقدمه

فصل ۱- مقدمه

۱-۱-پیشگفتار

در عصر حاضر با پیشرفت تکنولوژی، نیاز مبرم به اطلاعات در هر لحظه و هر مکان باعث ایجاد یک بازار بسیار بزرگ از محصولات الکترونیکی شده است. این وسایل الکترونیکی با صفحات لمسی مانند دستیار شخصی دیجیتال [1]، تلفن‌های همراه و ... با هدف بهبود تعامل انسان و ماشین جایگاه ویژه‌ای به خود اختصاص داده‌اند که باعث شده است بازار وسیعی از برنامه‌های کاربردی مختلف برای این وسایل نیز شکل بگیرد.

در این میان با توجه به روش‌های رایج ارتباطی بشر از جمله نوشتار، گفتار و زبان اشاره نوشتار همچنان به عنوان پرکاربردترین شیوه ثبت اطلاعات در زندگی روزمره محبوبیت خود را حفظ نموده است. گسترش و پیشرفت سیستم‌های با صفحات لمسی و از طرفی محبوبیت دستنوشته به عنوان یک رسانه ارتباطی در میان انسان‌ها، باعث گردیده تا توجه محققان به سیستم‌های تشخیص دستنوشته برای اینگونه وسایل جلب شود [۱].

سیستم‌های تشخیص دستنوشته به حوزه نویسه خوان نوری [2] مرتبطند. شناسایی نوری متون راهکاری است برای تبدیل متون تصویری به مستندات دیجیتالی قابل ویرایش توسط کامپیوتر و از لحاظ کاربردی برنامه‌ای است که با استفاده از روش‌های گوناگون هوش مصنوعی، پردازش تصویر و شناسایی الگو اسناد، مدارک، کتاب‌ها و سایر مکتوبات چاپی یا تایپ شده و یا دستنویس را به متن قابل ویرایش و قابل جستجو تبدیل می‌کند. چرخه سیستم‌های نویسه خوان نوری در شکل ۱-۱ نشان داده شده است.

شکل ۱-۱: چرخه کلی سیستم نویسه خوان نوری [۲]

[1] Personal Digital Assistant (PDA)
[2] Optical Character recognition (OCR)

تاریخچه تحقیقات در زمینه نویسه خوان نوری به زمان ظهور کامپیوترهای دیجیتالی باز می‌گردد و مطالعه در این زمینه همچنان ادامه دارد. .تاکنون در کشور ما و برخی کشورهای دیگر بر روی نویسه خوان‌های نوری فارسی و عربی کارهای ارزشمندی انجام شده است، اما می‌توان گفت که نویسه خوان‌های نوری زبان‌های لاتین، چینی و ژاپنی بسیار بیش از نویسه خوان های نوری فارسی به ایده آل نزدیک شده‌اند، هرچند این زمینه اخیراً توجه زیادی را به خود جلب کرده است [۳]. علت اصلی این امر پیچیدگی‌های منحصر به فرد نگارش فارسی است که این موضوع باعث شده تا تحقیقات به مراتب کمتری در زمینه نویسه خوان فارسی انجام شود و ازطرفی استفاده از نویسه خوان‌های زبان‌های نامبرده برای شناسایی متون فارسی امکان‌پذیر نباشد. اما در نهایت با غلبه بر چالش‌های مرتبط با زبان فارسی، با توجه به جمعیت بسیار زیاد فارسی زبانان و همچنین مشابهت الفبای زبان فارسی با زبان‌های دیگر از جمله عربی و اردو، ارائه سیستمی کارا برای تشخیص دستنوشته‌های فارسی/عربی برای وسایل الکترونیکی با صفحات لمسی می تواند با استقبال بسیار چشمگیری روبرو شود.

۱-۲- انواع سیستم‌های نویسه خوان نوری

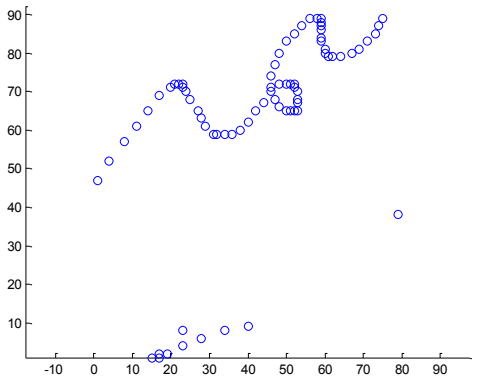
سیستم‌های نویسه خوان نوری را می‌توان از زوایای مختلفی تقسیم‌بندی کرد، بطور مثال می توان بر اساس نحوه نگارش به تایپی یا دستنوشته، برای متون تایپی بر اساس تک فونتی یا چند فونتی بودن، از لحاظ متصل بودن متن به حروف مجزا یا کلمات پیوسته و یا بر اساس نوع الگوی ورودی به برخط [۱] و برون خط [۲] و یا از لحاظ واحد شناسایی به حرف، رقم، کلمه تقسیم کرد.

۱-۲-۱-شناسایی برخط و برون خط

سیستم های نویسه خوان نوری بر اساس نوع الگوی ورودی به دو دسته سیستم‌های شناسایی برخط [۳] و برون خط تقسیم می‌شوند. در شناسایی برخط متن توسط قلم بر روی یک صفحه حساس نوشته می‌شود. این صفحه که در واقع شبکه‌ای از نقاط نزدیک به هم است توسط قلم تحریک شده و مختصات رقومی شده مسیر حرکت قلم به

[1] On-line
[2] Off-line
[3] Online handwriting recognition system (OHRS)

شکل مجموعه ای از نقاط دو بعدی به عنوان ورودی سیستم شناسایی استفاده می‌شود. نمونه‌ای از ورودی برخط در شکل ۱-۲ نشان داده شده است.

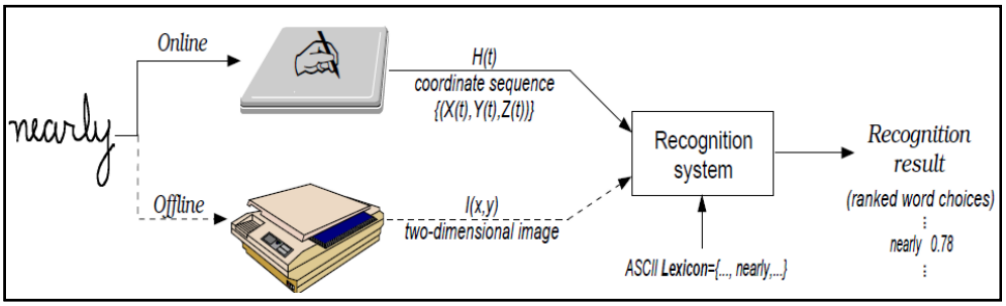


شکل ۱-۲: نمونه ای از یک کلمه دستنوشته برخط از پایگاه داده زیرکلمات برخط دانشگاه تربیت مدرس

در شناسایی برون خط برخلاف برخط، ورودی سیستم یک تصویر است که پیکسل‌های آن اطلاعات ثابتی را ارائه می‌دهند. نمونه‌ای از ورودی برون خط نیز در شکل ۱-۳ نشان داده شده است.

شکل ۱-۳: نمونه ای از یک تصویر دستنوشته برون خط از پایگاه داده ایران شهر

در شکل ۱-۴ این دو سیستم باهمدیگر به نمایش در آمده‌اند.



شکل ۱-۴: سیستم‌های تشخیص برخط و برون خط [۴]

مزیت شناسایی برخط در اختیار داشتن اطلاعات زمانی نظیر سرعت، شتاب، فشار و زمان‌های لمس یا عدم لمس صفحه است که سیستم‌های تشخیص دستنوشته برخط می‌توانند با استخراج این اطلاعات پویا و استفاده از آن‌ها دقت بیشتری نسبت به برون خط داشته باشند. بطور مثال زمان‌های لمس یا عدم لمس صفحه می‌تواند هر حرکت قلم[1] و ترتیب زمانی نوشته شدن آن‌ها را مشخص کند.

تنها مشکل عمده سیستم‌های تشخیص برخط آن است که نویسنده به تجهیزات خاصی برای نوشتن نیاز دارد که این تجهیزات در حال حاضر به اندازه قلم و کاغذ واقعی برای نوشتن راحت نیستند [۱]. هرچند صفحه یادداشتهای دیجیتالی[2] تا حدی این مشکل را برطرف کرده‌اند، بدین صورت که می‌توان بر روی آن‌ها کاغذ واقعی قرار داد.

بدلیل اینکه درحال حاضر تحقیقات در زمینه تشخیص دستنوشته برون خط موفقیت بسیار قابل قبولی در کاربردهای مختلف مثل خواندن آدرس‌های پستی، پردازش فرم، بررسی چک‌های بانکی و ... داشته است، این قضیه نیز باعث جلب بیشتر توجه محققان به زمینه تشخیص دستنوشته برخط شده است [۱].

۱-۳-رقومی کننده ها[3] و تبلت های گرافیکی

همانطور که قبلا گفته شد، در شناسایی برخط متن توسط قلم بر روی یک صفحه حساس رقومی کننده نوشته می‌شود و مختصات رقومی شده مسیر حرکت قلم به عنوان ورودی سیستم شناسایی استفاده می‌شود. ایده استفاده از قلم برای کامپیوتر برای اولین بار توسط کای در سال ۱۹۶۸ مطرح شد و از آن زمان به بعد نوآوری ها و پیشرفت‌های زیادی برای صفحات رقومی کننده رخ داد. برای مقایسه صفحات رقومی کننده از ویژگی‌های مختلفی می‌توان استفاده کرد مثل، درجه تفکیک پذیری، دقت، و نرخ نمونه‌برداری که معمولاً چیزی حدود ۲۰۰-۵۰ هرتز است [۱].

تکنولوژی بعدی که با ترکیب صفحات رقومی کننده با صفحات نمایشی بوجود آمد باعث ایجاد تبلت های گرافیکی شد که سطح بالایی از تعامل شبیه نوشتن و نقاشی کردن با استفاده از کاغذ و قلم معمولی را ارائه می کند. نمونه‌ای از یک تبلت گرافیکی و صفحه یادداشت دیجیتالی در شکل ۱-۵ نشان داده شده است.

[1] Stroke
[2] Digital Notepad
[3] Digitier

شکل ۱-۵: (سمت چپ) صفحه یادداشت دیجیتالی مدل CrossPad. (سمت راست) تبلت گرافیکی

در ساخت صفحات لمسی از دو فناوری مقاومتی و خازنی استفاده گسترده‌ای شده است که هر فناوری ویژگی‌های خاص خود را دارد. صفحات خازنی حساسیت بهتری نسبت به لمس با انگشت دارند ولی قادر به تشخیص ضربه توسط اشیا دیگر نیستند درحالی که صفحات مقاومتی قابلیت تحریک پذیری بیشتری دارند و می توانند توسط هر نوع قلم یا دست با دستکش نیز تحریک شوند. این فناوری هزینه کمی داشته و بیشتر مورد استفاده است. اخیراً قلم‌های الکترونیکی با فناوری اولتراسوند نیز مورد استفاده قرار گرفته‌اند [۵].

۱-۴-دستنوشته زبان فارسی

در سیستم‌های تشخیص دستنوشته هدف شناسایی کلمه است، اما کارهایی در زمینه تشخیص مواردی مانند ارقام، حروف مجزا و یا علائم مانند فرمول‌های ریاضی یا شیمی نیز صورت گرفته است. نوشتار زبان فارسی به شکل پیوسته و حاوی ۳۲ حرف است که از راست به چپ نوشته می‌شود. شکل تعدادی از حروف زبان فارسی بر اساس محل قرارگیریشان در کلمه که می تواند ابتدا، وسط، انتها و یا بشکل جداگانه باشد، می تواند تغییر کند، در زبان فارسی یک کلمه از یک یا چند زیر کلمه تشکیل می شود. زیرکلمه قسمتی از کلمه است که بدون درنظر گرفتن نشانه‌ها[1] و علامت‌های حروف می‌تواند در یک حرکت قلم روی کاغذ نوشته شود. منظور ما از حرکت، نوشته‌ایست

[1] Diacritic

که از زمان قردادن قلم تا زمان برداشتن آن ایجاد می شود و منظور از علامت‌ها، نقطه، سرکش دسته و یا مد حروف است که معمولاً به آن‌ها حرکات به تاخیر افتاده[1] نیز گویند و غالباً در نوشتار آخر از همه نوشته می‌شوند.

از جمله مواردی که باعث پیچیدگی نوشتار فارسی می‌شود می‌توان موارد زیر را نام برد [۶]:

- طبیعت پیوسته زبان فارسی
- وجود علامت‌های مد، همزه، دسته، سرکش و نقطه
- اشکال مختلف حروف در موقعیت‌های مختلف
- تغییر شکل حروف در نوشته پیوسته مثل حرف "ه" در "ها" و یا "ح" در "محمد"
- ابهام در شکل بعضی حروف به دلیل شکل‌های مختلف، مثل حرف "س" که می تواند بدون دندانه هم نوشته شود.
- شباهت کلی شکل حروف
- جابجا شدن موقعیت علامت حروف در کلمه بدلیل عدم دقت نویسنده

همانطور که قبلاً ذکر شد هدف سیستم‌های شناسایی دستنوشته شناسایی کل کلمه است، با این حال می‌توان با شکستن مسئله به مسائل کوچکتر کار را ساده تر کرده و به شناسایی زیرکلمه پرداخت. ساده‌سازی مسئله به شناسایی زیرکلمه باعث مطرح شدن این سوال می شود که چگونه بایستی زیرکلمات مربوط به یک کلمه را از کلمات دیگر متن دستنوشته تشخیص داد؟ برای حل این مشکل چندین راه حل می توان پیشنهاد داد که در ادامه به توضیح آنها می‌پردازیم.

- **استفاده از فاصله زمانی**: بدین معنی که مدت زمان سپری شده از زمان برداشتن قلم تا زمان گذاشتن مجدد آن روی صفحه اندازه گیری شده و درصورتی که این زمان از حد آستانه‌ای کمتر باشد، حرکت قلم جزء کلمه فعلی در نظر گرفته می شود و اگر این زمان از حد آستانه بیشتر باشد فرض بر آن گذاشته می شود که نوشتن کلمه به پایان رسیده است و حرکت فعلی قلم مربوط به کلمه بعدی است. این روش باعث می‌شود تا زیرکلمات و نشانه‌های هر کلمه به راحتی قابل تفکیک باشند و تنها دلیل پیدایش خطا در تفکیک حرکات هر کلمه بعلت رعایت نکردن فاصله زمانی توسط خود کاربر خواهد بود. عیب این روش فشاری است که به کاربر برای رعایت فاصله زمانی مابین حرکات قلم اعمال می‌شود.

[1] Delayed Stroke

- **استفاده از فاصله فضایی**: اختلاف فاصله فضایی حرکات قلم از یکدیگر می‌تواند به عنوان معیاری برای تشخیص زیرکلمات و علامت‌های یک کلمه بکار رود. در این حالت بطور مثال برای زبان فارسی که نوشتار در راستای افقی انجام می‌شود، اگر کمترین فاصله اقلیدسی مابین دو حرکت قلم در راستای افقی از حد آستانه‌ای کمتر باشد، می‌توان حرکات را مربوط به یک کلمه دانست و اگر این فاصله از حد آستانه تجاوز کرد حرکت فعلی قلم را مربوط به کلمه‌ی بعدی در نظر گرفت. عیب این روش نیز محدودیتی است که برای نویسنده به منظور رعایت فاصله بین کلمات اعمال می‌شود.
- **استفاده از فرهنگ لغت**: با استفاده از یک فرهنگ لغت، با بررسی ترکیب‌های مختلف زیرکلمات مجاور با یکدیگر می‌توان زیرکلمات مربوط به هر کلمه را تشخیص داد. مزیت این روش نسبت به دو روش قبل عدم ایجاد محدودیت برای نویسنده است. هرچند در این حالت مشکل دیگری که مطرح می شود نحوه تعیین حرکات زیرکلمات از حرکات علامت‌ها خواهد بود.

در ادامه این پایان‌نامه تمرکز ما بر روی شناسایی زیرکلمات خواهد بود که در این زمینه دو رویکرد متفاوت مطرح می‌شود که در ادامه به آن می‌پردازیم.

۱-۵-دو رویکرد کلی در زمینه شناسایی زیرکلمه

برای تشخیص زیرکلمه دو رویکرد متفاوت وجود دارد. رویکرد اول یا رویکرد کلی‌نگر[1] آن است که واحد شناسایی ما خود زیرکلمه باشد. یعنی هر زیرکلمه به عنوان یک الگو درنظر گرفته شود و پارامترهای استخراجی نماینده کل زیرکلمه هستند. در این رویکرد هر کلمه به عنوان یک نهاد غیرقابل تفکیک درنظر گرفته می‌شود. درنتیجه تصمیم گام شناسایی تنها اختصاص یک برچسب از یک مجموعه از فرهنگ لغت به کلمه‌ی تست خواهد بود. رویکرد دوم یا رویکرد جزئی‌نگر[2] آن است که شناسایی زیرکلمه با شناسایی واحدهای معنادار سازنده آن صورت گیرد.

رویکرد کلی نگر ایجاب می کند تا یک پایگاه داده از الگوهای مورد شناسایی که درواقع زیرکلمات هستند را داشته باشیم. در این حالت برای شناسایی، هر زیرکلمه تست ورودی بایستی با تمام الگوهای داخل پایگاه دانش مقایسه شود.

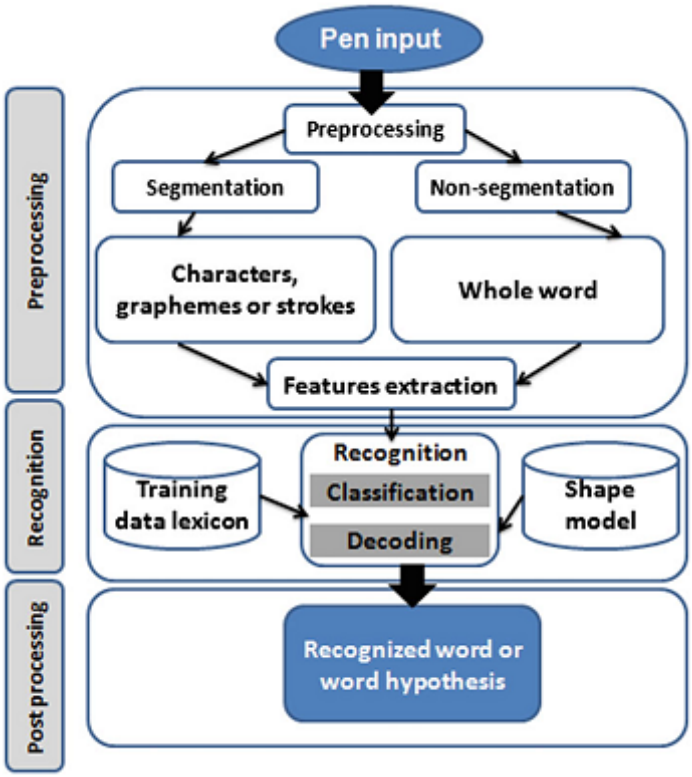
[1] Holistic approach
[2] Analytical approach

نقطه ضعف این رویکرد آن است تعداد زیرکلمات قابل شناسایی محدود به الگوهای موجود در پایگاه دانش است و اگر الگوی ورودی در پایگاه دانش نباشد، شناسایی الگوی ورودی امکان‌پذیر نیست. عیب دیگر این رویکرد آن است که اضافه‌کردن الگوهای بیشتر به پایگاه دانش باعث بزرگ شدن بیش از اندازه پایگاه دانش می‌شود، که در نتیجه با بالا رفتن تعداد کلاس‌های خروجی قاعدتاً دقت شناسایی کمتر و به مراتب زمان شناسایی بیشتر می‌شود. درنتیجه این رویکرد برای برنامه‌های با فرهنگ لغت گسترده چندان کاربردی نیست.

در مقابل مزیت رویکرد جزئی‌نگر آن است که به تقسیم مسئله شناسایی کلمه به چندین زیرمسئله مکمل یکدیگر از شناسایی حروف یا حرکت‌ها می پردازد. این مزیت باعث می شود تا رویکردهای جزئی نگر برای مسائل شناسایی کلمه با فرهنگ لغات بسیار گسترده یا حتی باز مناسب‌تر باشند، هرچند بسیاری از مسائل در رویکرد جزئی‌نگر حل نشده باقی مانده‌اند. بطور مثال سیستم‌های بالا به پائین (با قطعه بندی ضمنی) نرخ شناسایی نسبتا بالایی دارند، هرچند هنوز به کاربردهای با فرهنگ لغت کوچک محدودند. بالعکس سیستم‌های با رویکرد قطعه‌بندی آشکار که برای تعامل با مسائل فرهنگ لغت باز طراحی شده‌اند، هنوز نتوانسته‌اند به نرخ شناسایی نسبتا بالایی دست یابند [۱].

۱-۶-تحلیل اجزای سیستم‌های تشخیص دستنوشته برخط

در این قسمت ما به تحلیل بخشهای مختلف یک سیستم تشخیص دستنوشته برخط همه منظوره می‌پردازیم که در شکل ۱-۶ نمایی از بخش‌های مختلف آن نشان داده شده است. هر چند سیستم نشان داده شده یک سیستم متعارف نیست، اما به نوعی گویای تمام سیستم‌های ارائه شده تاکنون است. در این مدل، سه جز اصلی وجود دارد که شامل گام‌های پیش پردازش، استخراج ویژگی، قطعه‌بندی، شناسایی و پس‌پردازش هستند. گام‌های نامبرده در تمام سیستم‌های تشخیص دستنوشته برخط الزاما وجود ندارند اما همیشه چند مورد از آن‌ها همیشه وجود دارد بنابراین ممکن است در بعضی از تحقیقات گام‌های بیشتری ذکر شود و در بعضی دیگر ممکن است بعضی از گام‌ها حذف شده باشند. در ادامه به بررسی وظیفه گام‌های مختلف سیستم دستنوشته برخط می‌پردازیم [۱].



شکل ۱-۶: اجزای سیستم تشخیص دستنوشته برخط همه منظوره معرفی شده در [۱]

۱-۶-۱- پیش پردازش

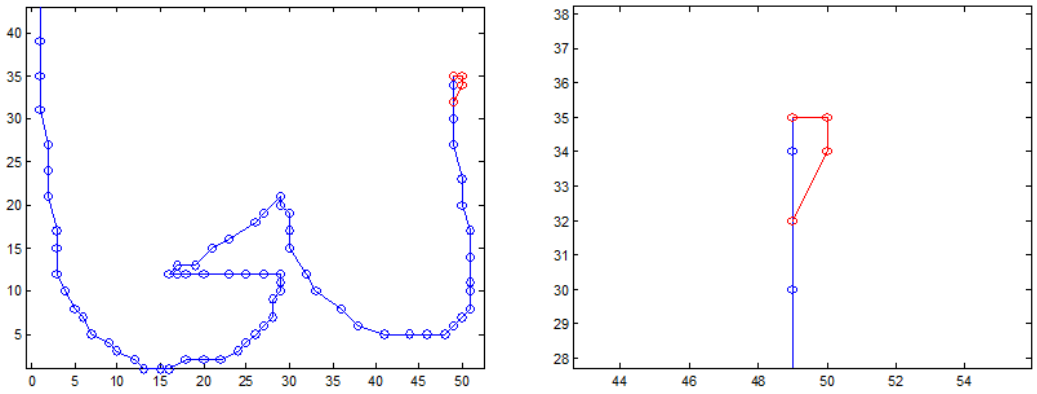
همواره پیش‌پردازش یکی از گام‌های پایه در تشخیص دستنوشته است و برای دستیابی به نرخ شناسایی بالاتر همواره ضروری است. معمولاً قبل از هر عملیات شناسایی، داده‌های ورودی بایستی پیش‌پردازش شوند. هدف اصلی گام پیش‌پردازش کاهش نویز، حذف نقوص سخت‌افزار و همینطور از بین بردن اثر لرزش‌های حین نوشتن است.

در بعضی سیستم‌های تشخیص دستنوشته گام مهم دیگری که بعد از گام‌های همواسازی، فیلترینگ و نمونه برداری مجدد و نرمالسازی در مرحله پیش‌پردازش انجام می‌شود، تخمین خط زمینه[1] است. برای سیستم دستنوشته برخط، خط زمینه اساساً برای تعیین نشانه‌ها، قطعه‌بندی و استخراج ویژگی بکار می‌رود.

یکی دیگر از عملیات‌هایی که ممکن است در گام پیش‌پردازش انجام گیرد حذف قلاب[2] است. در نوشتار برخط به دلیل حساسیت زیاد بعضی صفحات به جنس قلم، با نزدیک یا دور کردن قلم از صفحه در ابتدا و انتهای حرکت

[1] Baseline
[2] Dehooking

قلم، ممکن است تعدادی از نقاط صفحه ناخواسته تحریک شوند و شکلی شبیه به قلاب ایجاد شود. در شکل ۱-۷ نمونه ای از یک قلاب ایجاد شده در یک داده برخط نشان داده شده است.



شکل ۱- ۷: شکل سمت چپ. داده ورودی برخط با هوک ابتدایی. شکل سمت راست. نمای بزرگ شده از هوک که با رنگ قرمز مشخص شده است

۱-۶-۲- استخراج ویژگی

وظیفه استخراج ویژگی آن است تا با محاسبه و استخراج بهترین مشخصه‌های سیگنال ورودی باعث کوچک شدن الگوهای ورودی و درنتیجه یک کلاسه‌بندی بهتر شود.

ویژگی‌های قابل استفاده برای تشخیص دستنوشته برخط را می‌توان به ویژگی‌های شکلی، ساختاری، آماری و ویژگی‌های حوزه تبدیل مثل تبدیل‌های فوریه، موجک و ... تقسیم کرد.

ویژگی‌های شکلی، رایج‌ترین نوع ویژگی استفاده شده در تحقیقات انجام شده است. این ویژگی‌ها نشان‌دهنده شکل دستنوشته هستند. به عنوان مثال زاویه بین نقاط متوالی دستنوشته یک ویژگی شکلی است.

ویژگی‌های ساختاری مربوط به ساختارهای نوشته مثل وجود یا عدم وجود حلقه، نقاط کمینه و بیشینه، نقاط تیز و موارد مشابه می شود. برای افزایش نرخ بازشناسی معمولاً از ویژگی‌های شکلی به همراه ویژگی‌های ساختاری استفاده می‌شود. ویژگی‌های شکلی و ساختاری را می‌توان به دو دسته محلی و سراسری تقسیم کرد. در صورتی که برای استخراج ویژگی از کل نوشته استفاده شود، سراسری و در غیر این صورت محلی است. به عنوان مثال، جهت بردار متصل‌کننده نقطه اول و آخر یک حرف یک ویژگی سراسری شکلی است.

ویژگی‌های حوزه‌های تبدیل اطلاعات ملموسی ندارند و به همین دلیل کمتر در تشخیص دستنوشته برخط مورد استفاده قرار گرفته‌اند. از جمله ویژگی‌های موفق مورد استفاده حوزه تبدیل می‌توان توصیفگر ممان‌های هو[1] [۷] که ویژگی آن‌ها عدم حساسیت به اندازه و چرخش است را نام برد.

ویژگی‌های آماری نیز، ویژگی‌هایی هستند که با بررسی آماری نمونه‌های متعدد دستنوشته و تجزیه و تحلیل آن‌ها بدست می‌آیند. از ویژگی‌های آماری بیشتر در تشخیص دستنوشته برون خط استفاده می‌شود.

۱-۶-۳- قطعه بندی

به روش‌های گوناگونی که به منظور بدست آوردن اجزا پایه بامعنی برای پردازش در الگوریتم‌های شناسایی انجام می‌شود قطعه‌بندی گویند. قطعه‌بندی در رویکرد جزئی‌نگر بر اساس اینکه گام قطعه‌بندی قبل از گام استخراج ویژگی و شناسایی کلاس انجام شود و یا اینکه گام شناسایی کلاس به هدایت گام قطعه‌بندی بپردازد، ما به وضعیت اول رویکرد جزئی‌نگر با قطعه‌بندی خارجی یا قطعه‌بندی آشکار و به وضعیت دوم رویکرد جزئی‌نگر با قطعه‌بندی داخلی یا قطعه‌بندی ضمنی گوییم.

۱-۶-۴- شناسایی

گام شناسایی سیستم در واقع بکارگیری الگوریتم‌های کلاسه‌بندی است. در این گام از داده‌های آموزشی برای تشخیص واحدهای منحصرد بفرد پایه استفاده می‌شود. کار شناسایی شامل مقایسه الگوی تست با الگوی مرجع هر کلاس که نماینده کلمات یک فرهنگ لغت‌اند و همین طور اندازه‌گیری میزان شباهت بین الگوی تست و الگوی کلاس مرجع است. میزان شباهت الگو بعداً برای آن که بفهمیم کدام الگو بیشترین شباهت را با الگوی ناشناخته دارد استفاده می‌شود. پیاده‌سازی گام شناسایی در سیستم‌های قبلی به روش‌های مختلفی انجام شده است از جمله: درخت تصمیم‌گیری[2]، برنامه‌نویسی پویا[3]، انطباق الگو[4]، مدل مخفی مارکوف[5]، شبکه عصبی[6]، K-نزدیکترین

[1] Hue moments
[2] Decision Tree
[3] Dynamic Programming
[4] Template Matching
[5] Hidden Markov Model
[6] Neural Network

همسایه[1]، و یا ترکیبی از روش‌ها. معمولاً فرآیند شناسایی لیستی از کلمات پیشنهادی و یا فقط یک کلمه یا حرف را تهیه می‌کند. با کمک نوعی از منبع دانش به شکل مدل زبان[2]، بهبودهایی در شناسایی می‌توان انجام داد. یک مدل زبان می‌تواند یک فرهنگ لغت باشد، که یک کتابخانه یا لیست از کلمات ممکن برای شناسایی است، یا کلماتی که اجازه ورود به سیستم شناسایی را دارند، یا می‌تواند یک سری خواص آماری یا ساختاری از زبان داده شده باشند.

۱-۷- هدف از تحقیق

در تحقیق پیش رو هدف ما آن است تا به بررسی و ارائه سیستمی برای تشخیص دستنوشته برخط فارسی بپردازیم. برای این کار در ابتدا برای غلبه بر پیچیدگی های دستنوشته فارسی بعد از اعمال یک سری پیش پردازش بر روی زیرکلمه ورودی از یک تکنیک ساده‌سازی برای نمایش نقاط ورودی بشکلی مستطیل وار بوسیله دنباله‌ای از خطوط افقی و عمودی استفاده شده است. سپس از یک روش انطباق الگو که با تکنیک برنامه‌نویسی پویا به محاسبه فاصله اصلاح مینیمم می‌پردازد، برای شناسایی بدنه حروف تشکیل‌دهنده زیرکلمه استفاده می‌شود. بدین منظور از پایگاه داده زیرکلمات برخط دانشگاه تربیت مدرس [۸] استفاده شده است.

۱-۸- ساختار پایان نامه

در فصل ۲ به مرور مهمترین کارهای انجام شده در زمینه تشخیص دستنوشته برخط عربی و فارسی می‌پردازیم. در موارد ذکر شده سعی بر آن بوده تا انواع روش‌ها و ایده‌های مختلفی که تاکنون برای شناسایی دستنوشته مورد استفاده قرار گرفته در یک دسته‌بندی مناسب تشریح شود.

در فصل ۳ بطور مختصر به تئوری‌های مرتبط و مورد استفاده در این تحقیق پرداخته شده است. برای جزئیات بیشتر در مورد این تئوری‌ها می توان به مراجع انتهای تحقیق مراجعه نمود، هرچند سعی شده تا اطلاعات ذکر شده در این فصل برای درک کامل روش پیشنهادی کاملاً کافی باشد. در مواردی که تئوری مبحثی در این فصل ذکر نشده باشد، سعی شده در جای خود به تفصیل به توضیح تئوری مربوطه پرداخته شود.

[1] K-nearest neighbor
[2] Language model

در فصل ۴ روش مورد استفاده در این پایان‌نامه به طور کامل توضیح داده شده است. این فصل برای درک صحیح‌تر روند کار به بخش‌های مختلفی از جمله آماده‌سازی داده‌ها، پیش‌پردازش داده‌ها، ساده‌سازی یا استخراج ویژگی‌ها، تولید پایگاه دانش و درنهایت تشخیص زیرکلمه تقسیم شده است.

نتایج شبیه‌سازی روش پیشنهادی و تحلیل این نتایج در فصل ۵ ذکر خواهد شد. هرچند بدلیل تفاوت‌های بسیار زیاد تحقیقات انجام شده مخصوصاً تفاوت در پایگاه داده مورد استفاده، مقایسه نتایج با کارهای گذشته چندان معنادار نیست اما نتایج کارهای گذشته صرفا جهت اطلاع در این فصل ذکر شده اند. در پایان با ذکر ویژگی‌ها، نقاط قوت و ضعف روش، نتیجه‌گیری و پیشنهاداتی در جهت بهبود روش در فصل ۶ ارائه خواهد شد.

فصل ۲

مروری بر کارهای گذشته

فصل ۲- مروری بر کارهای گذشته

در این فصل به مرور تعدادی از مهمترین تحقیقات انجام شده در زمینه تشخیص دستنوشته برخط فارسی و زبان‌های دیگر با الفبای مشابه مانند عربی و اردو می‌پردازیم. تفاوت در واحدهای شناسایی، نحوه تعامل با نشانه‌ها، رویکرد مورد استفاده، حذف و اضافه شدن گام‌های مختلف و ... همه و همه باعث شده است تا تنوع زیادی در تحقیقات انجام شده در زمینه‌های مختلف سیستم‌های تشخیص دستنوشته برخط ایجاد شده و کار دسته‌بندی جامع این تحقیقات کار ساده‌ای نباشد.

در این فصل ما ابتدا به بررسی چند تحقیق انجام شده با رویکرد کلی‌نگر می‌پردازیم تا خواننده بتواند درک بهتری از نحوه کارکرد و کارایی سیستم‌های مبتنی بر این رویکرد بدست آورد.

در سیستم‌هایی که اقدام به قطعه بندی آشکار می‌کنند گام قطعه بندی در نتیجه نهایی تاثیر مهمی دارد، به همین دلیل در این زمینه نیز کارهایی صورت گرفته که چند مورد از آن نیز مطرح خواهد شد.

تحقیقات در زمینه‌های تشخیص حروف جدا یا ایزوله[1] نیز از دیگر حوزه‌های مورد علاقه محققان در زمینه تشخیص دستنوشته برخط است، به همین دلیل به بررسی چند مورد از تحقیقات انجام شده در این حوزه نیز خواهیم پرداخت و درنهایت به بررسی سیستم‌های با رویکرد جزئی‌نگر می‌پردازیم.

۲-۱- روش‌های تشخیص زیرکلمه کلی نگر

همانطور که قبلاً گفته شد در رویکرد کلی‌نگر هر زیرکلمه به عنوان یک واحد غیر قابل تجزیه درنظر گرفته می‌شود. مزیت این رویکرد آن است که باعث می‌شود تا گام قطعه‌بندی و پیچیدگی‌های آن از سیستم حذف شود. حذف گام قطعه‌بندی از آن جهت حائز اهمیت است که عموما خطای رخ داده در این گام مستقیماً به گام شناسایی وارد می‌شود و غالبا امکان تصحیح خطاهای رخ داده در گام قطعه‌بندی در گام‌های بعدی وجود ندارد.

Saabni و El-Sana [۹] در روش کلی‌نگر خود از ویژگی‌های هندسی سراسری به عنوان ورودی کلاسه‌بند سیستم که با تکنیک پیچش زمانی پویا[2] کار می‌کند، استفاده نموده‌اند. این کلاسه‌بند به تعیین و مرتب کردن مدل‌های آموزشی (کاندید) که تطابق بیشتری با رشته ورودی دارند می‌پردازد. سپس K کاندید با بالاترین امتیاز به یک

[1] Isolate
[2] Dynamic time warping (DTW)

کلاسه‌بند خطی ساده دیگر که بر اساس شکل محتوا کار می‌کند ارسال می‌شوند تا شناسایی نهایی کلمه انجام شود. در حین فرایند شناسایی از چند فیلتر بصورت سلسله مراتبی برای کاهش فضای جستجو نیز استفاده شده است. این روش بر روی ۶۰۰ جزء کلمه نوشته شده توسط ۶ نفر آزمایش شده و نرخ شناسایی چیزی بین ۸۶ تا ۹۰ درصد گزارش شده است.

رضوی و کبیر [۱۰] برای بازشناسی زیرکلمات فارسی، ابتدا کل پایگاه داده را با توجه به علامت‌های زیرکلمات گروه‌بندی کرده‌اند. سپس با این فرض که ترتیب قرار دادن علامت‌ها از نظر مکان رعایت شود و نویسنده‌ها محدودیت‌های مربوط به شکل علامت‌ها را نیز رعایت کرده باشند، زیرکلمات با توجه به ترتیب انواع علامت‌ها دسته‌بندی می‌شوند. به عنوان مثال زیرکلماتی که به ترتیب یک نقطه بالا و دو نقطه پایین دارند در یک گروه قرار می‌گیرند. برای شناسایی نقطه از اندازه آن استفاده می‌شود و برای تشخیص علامت‌های دیگر مشابه با روش تشخیص بدنه زیرکلمات عمل می‌شود با این تفاوت که علامت به ۱۰ نقطه نرمالیزه می‌شود.

برای تشخیص بدنه، زیرکلمه به ۵۰ نقطه نرمالیزه می‌شود و زاویه بردارهای متصل کننده نقاط به دست می‌آید، مجموع وزن‌داری از مجموع قدرمطلق اختلاف زوایا و مجموع قدرمطلق فاصله اقلیدسی بین نقاط به عنوان معیار فاصله در نظر گرفته شده است. این روش بر روی پایگاه داده زیرکلمات برخط دانشگاه تربیت مدرس آزمایش شده است و نرخ بازشناسی برای گزینه اول برابر با ۷۴/۹۵ درصد است [۱۰].

در تکمیل کار قبلی، سیستمی برای بازشناسی کلمات فارسی پیشنهاد شده است که بعد از بدست آوردن کاندیدهای هر زیرکلمه، امتیاز هر زیرکلمه برابر با کمترین فاصله تقسیم بر فاصله آن زیرکلمه درنظر گرفته شده است و امتیاز کلمه از حاصل ضرب امتیاز زیرکلمات کاندید به دست می‌آید، با توجه به فرهنگ کلمات فارسی موجود، کلمات کاندید غیر موجود حذف می‌شوند. این سیستم روی یک متن که توسط یک نفر نوشته شده آزمایش شده است که دقت بازشناسی آن در سطح کلمه ۱۳۱ از ۱۵۰ و در سطح زیرکلمه ۳۲۹ از ۳۵۳ است [۱۱].

فرجی و همکاران [۱۲] برای بازشناسی زیرکلمات پایگاه داده تربیت مدرس [۸]، ابتدا آن‌ها را بر اساس علامت گروه‌بندی کرده‌اند، در این گروه‌بندی با توجه به این که نویسنده‌ها ترتیب علامت‌ها را رعایت نمی‌کنند، فقط وجود علامت‌های مختلف درنظر گرفته شده است به عنوان مثال تمام زیرکلمات دارای یک سرکش و دو نقطه بالا در یک گروه قرار می‌گیرند و ترتیب وقوع نشانه‌ها اهمیت ندارد.

مراحل پیش‌پردازش بدنه اصلی شامل حذف قلاب، هموارسازی، نرمالیزاسیون و همسان‌سازی کادر نوشته است. از ویژگی جهت بردار ابتدا به انتهای نوشته که یا رو به پایین و یا رو به بالا است برای زیرطبقه‌بندی قبل از طبقه‌بندی نهایی استفاده می‌شود، سپس زاویه بردارها به ۸ جهت اصلی رقمی شده و به عنوان ویژگی در کلاسه‌بند مدل مخفی مارکوف استفاده می‌شود. نتیجه این روش برای بهترین پارامترها، ۸۲/۴۳ درصد است [۱۲].

کار قبلی با استفاده از ۳ مرحله کلاسه‌بند شبکه عصبی RBF و افزودن ویژگی جهت رقمی شده بردارهای متصل‌کننده نقاط تیز و بیشینه محلی متوالی ۸۱/۳۱ درصد گزارش شده است [۱۳].

۲-۲- قطعه‌بندی

Daifallah و همکاران [۱۴] ابتدا در گام پیش‌پردازش به هموارسازی، خوشه بندی نقاط و حذف قلاب پرداخته‌اند و از یک فیلتر پایین گذر برای هموارسازی استفاده کرده‌اند. سپس الگوریتمی برای قطعه‌بندی حرکت‌ها به حروف پیشنهاد نمودند که قادر است تمام نقاط قطعه‌بندی صحیح را شناسایی کند ولی ممکن است بین ۵۰ تا ۶۰ درصد نقاط قطعه‌بندی اضافی[1] نیز پیشنهاد دهد. این الگوریتم کار خود را در چهار مرحله: قطعه‌بندی اختیاری، بهبود قطعه‌بندی، اتصال نقاط مجاور و درنهایت تعیین نقاط قطعه‌بندی انجام می‌دهد. مبنای کار الگوریتم قطعه‌بندی بر این اساس است که در نوشتار عربی در محل اتصال حروف پاره خطی افقی از راست به چپ وجود دارد. در این روش از ممانهای Hue که شامل هفت جزء تغییرناپذیر نسبت به چرخش، انتقال و مقیاس[2] هستند به عنوان ویژگی استفاده شده است. درنهایت از مدل مخفی مارکوف برای شناسایی حروف استفاده شده است. این روش بر روی ۱۵۰ کلمه که حاوی ۷۲۰ حرف بوده اند تست شده و نرخ شناسایی ۸۵/۳ تا ۹۲/۶ درصد برای کلمات و ۸۸/۸ تا ۹۷/۲ برای حروف گزارش شده است.

در [۱۵] potrus و همکاران در ابتدا برای ساده‌تر شدن گام قطعه‌بندی پس از پیش‌پردازش هموارسازی و نرمالیزاسیون، یک طرح مستطیل وار از دستنوشته می‌سازند تا تشخیص نواحی قطعه بندی آسان‌تر شود. در مرحله نرمالیزاسیون ابتدا زاویه خط واصل بین هر دو نقطه متوالی به یکی از جهت‌های ۸ گانه تبدیل می‌شود که این بردارها مجموعه S را تشکیل می‌دهند، این مجموعه از مجموعه‌های S_j تشکیل شده است که $j = \{1,2, \dots ,8\}$

[1] Over Segmentation
[2] Scale

شماره مربوط به جهت بردار بین نقاط متوالی دستنوشته است. هر مجموعه S_j خود از زیرمجموعه های S_{ij} تشکیل شده که شامل بردارهای هم جهت متوالی است.

مجموعه های S_j که احتمال رخداد آنها در کلمه کمتر از λ باشد حذف می شوند:

$$\lambda = \frac{1}{10} log\left(\frac{N}{2}\right) \qquad (۱-۲)$$

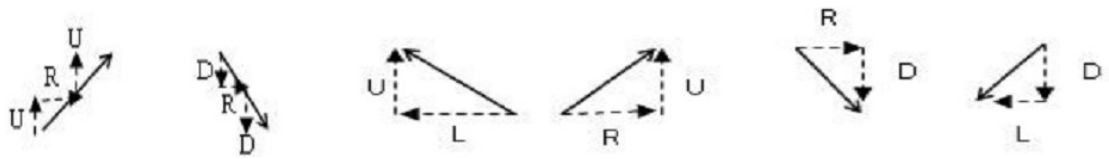
که N تعداد اعضای مجموعه S است. سپس در مرحله اصلاح، مجموعه‌های S_{ij} که کمتر ازϵ عضو دارند هم حذف می‌شوند. مقدار $\epsilon = 2$ بصورت تجربی بدست آمده است.

هشت جهت شکل ۲-۱ با توجه به جهت حرکت در هر نقطه تعریف می شوند و دستنوشته با این تعریف کد می‌شود.

$$g \in W_{(U,D,L,R)} : g = \begin{cases} U & \Delta x = 0, \Delta y > 0 \\ D & \Delta x = 0, \Delta y < 0 \\ L & \Delta x > 0, \Delta y = 0 \\ R & \Delta x < 0, \Delta y = 0 \\ RU & \Delta x < 0, \Delta y > 0 \\ RD & \Delta x < 0, \Delta y < 0 \\ LU & \Delta x > 0, \Delta y > 0 \\ LD & \Delta x > 0, \Delta y < 0 \end{cases}$$

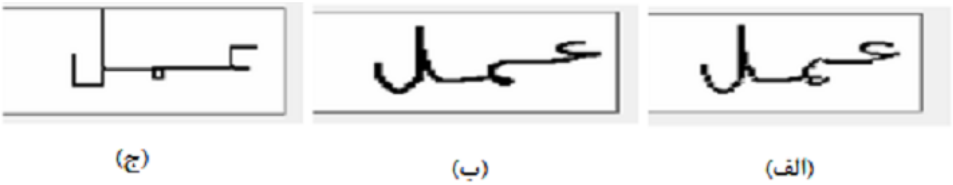
شکل ۲-۱:کدهای جهتی در [۱۵]

سپس مطابق شکل ۲-۲ هر جهت قطری به دو جهت اصلی تبدیل می‌شود، جهت‌های قطری متوالی هم مطابق شکل سمت چپ کد می شوند.



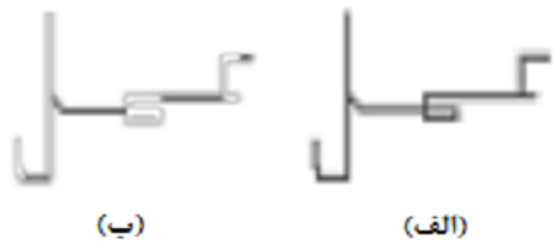
شکل ۲-۲: تبدیل جهت های قطری به اصلی [۱۵]

به این ترتیب دستنوشته به صورت کدی شامل جهت‌های اصلی در می‌آید و در مرحله بعد جهت‌های یکسان متوالی که طول کمتر از ۲ (تجربی) دارند با جهت همسایه‌ای که طول بیشتری دارد جایگزین می‌شود. حاصل این مراحل در شکل ۲-۳ نشان داده شده است.



شکل ۲- ۳: (الف) دستنوشته اصلی، (ب) پس از هموارسازی، (ج) شکل مستطیل‌وار [۱۵]

همان طور که دیده می‌شود، حاصل این عملیات شکلی است که از قسمت‌هایی تشکیل شده است که فقط یک جهت دارند و تشخیص حرف و خط پیوند در آن بسیار راحت‌تر است. تمام حروف عربی بعد از این که به این شکل در آورده شوند در جهت عمودی دارای نقاط مشترک هستند از این رو هر قسمتی که در جهت عمودی با هیچ نقطه دیگری اشتراک نداشته باشد خط پیوند در نظر گرفته می‌شود و بقیه قسمت‌ها هم حرف در نظر گرفته می‌شوند و نقطه وسط هر قطعه به عنوان نقطه قطعه‌بندی در نظر گرفته می‌شود. خروجی این مرحله برای یک نمونه در شکل ۲-۴ نشان داده شده است.



شکل ۲-۴: خروجی مرحله خط پیوند و حرف. (الف) شکل مستطیل وار زیرکلمه، (ب) خط پیوندهای تشخیص داده شده (قسمت های پر رنگ) [۱۵]

در این حالت قسمتی از حرف اول و آخر هم ممکن است به عنوان خط پیوند شناسایی شوند که به خاطر شکل بعضی از حروف عربی است، برای حذف خط پیوند اشتباه در حرف اول، اگر در بازه مربوط به خط پیوند شناسایی شده، تغییرات عمودی نداشته باشد، حرف وگرنه خط پیوند است. برای حذف خط پیوند اشتباه در حرف آخر، در هر نقطه از خط پیوند تشخیص داده شده زاویه بردار واصل آن نقطه تا آخرین نقطه دستنوشته کوچک‌تر از ۷۰ درجه باشد و مختصات عمودی یا کاملاً صعودی و یا کاملاً نزولی باشد.

نتیجه کلی این روش ۹۶/۲ درصد قطعه‌بندی صحیح بوده است که بر روی کلماتی که توسط ۲۰ نویسنده نوشته شده و شامل ۲۳۰۰ حرف است آزمایش شده است [۱۵].

عراقی و عبدالعظیم [۱۶] بعد از پیش‌پردازهای هموارسازی و نرمالیزاسیون، روشی برای پیداکردن محل خط زمینه ارائه کرده‌اند که در تخصیص علامت‌ها به حروف استفاده می‌شود، سپس برای قطعه‌بندی زیرکلمه‌های پایگاه داده ADAB [۱۷] به نویسه‌هایی که شامل حرف یا قسمتی از حرف هستند یک روش مستقل از خط زمینه ارائه کرده است که شامل مراحل زیر است:

نقاطی که زاویه خط متصل‌کننده آن نقطه به بعد با محور افقی کمتر از ۲۲ درجه باشد به عنوان نقاط قطعه‌بندی در نظر گرفته می‌شوند. سپس نقاطی که بالای آن با زاویه $\pm$۲۰ درجه نوشته وجود دارد حذف می‌شوند. نقاط متوالی به عنوان یک فاصله قطعه‌بندی در نظر گرفته شده و فواصل قطعه‌بندی که فاصله کمتر از ۵ نقطه دارند به هم متصل می‌شوند تا یک فاصله قطعه‌بندی بزرگ‌تر تشکیل دهند. هر نقطه‌ای که زاویه آن با نقطه بعد بین ۲۱۰ تا ۳۳۰ درجه باشد، رو به پایین و درصورتی که بین ۳۰ تا ۱۵۰ درجه باشد، رو به بالا فرض می‌شود، مراحل ذکر شده برای نقاط قطعه‌بندی برای نقاط رو به پایین و رو به بالا هم انجام می‌شود. فواصل رو به پایین و رو به بالا که کمتر از ۳ نقطه داشته باشند به عنوان نویز تلقی شده و حذف می‌شوند، نقاط وسط فواصل باقی مانده به ترتیب، مرکز رو به پایین و مرکز رو به بالا نامیده می‌شوند [۱۶].

نقاط قطعه‌بندی که قبل از آن یک مرکز رو به بالا (یا شروع زیرکلمه) و بعد یک مرکز رو به پایین و بعد از آن یک مرکز رو به بالا و سپس یک مرکز رو به پایین (یا انتهای زیرکلمه) باشد، قبول شده و بقیه حذف می‌شوند. همچنین نقاط قطعه‌بندی که قبل از آن یک مرکز رو به بالا و بعد از آن مرکز رو به پایین باشد به عنوان نقاط تپه‌ای شناخته شده و حذف می‌شوند، این نقاط روی دندانه‌های حروف قرار دارند. سپس فواصل قطعه‌بندی که در روی حلقه حروف قرار دارند با شمارش نقاطی که در ادامه نوشته زیر آن‌ها قرار دارد و قرار دادن یک مقدار آستانه حذف می‌شوند. در انتها هم فواصل قطعه‌بندی که در ابتدا یا انتهای زیرکلمه قرار دارند حذف می‌شوند [۱۶].

۲-۳- تشخیص حروف جداگانه

در سیستم‌های تشخیص دستنوشته دو مسئله اساسی وجود دارد که تمام راه‌حل‌های پیشنهادی را به چالش می‌کشد. مسئله اول نامشخص بودن تعداد حروف بکار رفته در زیرکلمه است. از لحاظ تئوری محدودیتی در تعداد حروف یک کلمه وجود ندارد، هر چند در عمل تعداد حروف یک کلمه در زبان فارسی از تعداد انگشتان دو دست تجاوز نمی‌کند. مسئله دیگر محل اتصال حروف به یکدیگر است. این دو مسئله در تشخیص حروف جداگانه وجود

ندارند و به همین دلیل سیستم‌های تشخیص حروف جداگانه نسبت به تشخیص دستنوشته پیوسته به مراتب ساده‌تر و کارایی بهتری دارند.

قدس و کبیر [۱۸] روشی مبتنی بر درخت تصمیم‌گیری و با تاکید بر استخراج ویژگی ارائه نموده‌اند. در روش ارائه شده بر اساس پیچیدگی هر حرف تا حداکثر ۲۴ ویژگی استخراج و سپس تعلق هر حرف به ۹ گروه مشخص می‌شود. درخت تصمیم در این روش پس از آموزش، گروه‌بندی را با دقت ۹۴ درصد انجام می‌دهد.

سلیمانی باغشاه و همکاران [۱۹] از ترکیب رویکردهای فازی، ساختاری و یادگیری با استفاده از شبکه عصبی برای شناسایی حروف جداگانه فارسی استفاده کرده‌اند. در ابتدا یک مرحله پیش‌پردازش روی ورودی انجام می‌شود تا نقاطی که فاصله کمی با نقاط مجاور دارند حذف شوند. سپس قطعه‌بندی حرف در سه مرحله انجام می‌شود، حلقه‌ها با پیداکردن محل تقاطع دستنوشته به عنوان یک قطعه در نظر گرفته می‌شوند، برای قطعه‌بندی قسمت‌های باقیمانده، نقاطی که بیشینه محلی هستند و تغییر زاویه در آن‌ها زیاد است همچنین نقاط عوض شدن جهت تقعر که تغییر زاویه در آن‌ها زیاد است به عنوان محل قطعه‌بندی معرفی می‌شوند.

ویژگی‌های استخراج شده از هر قطعه شامل این موارد است: جهت بردارهای متصل‌کننده ابتدا به انتها، ابتدا به مرکز ثقل و انتها به مرکز ثقل قطعه، میزان انحنای قطعه که از اختلاف زاویه بین بردارهای ابتدا و انتها به مرکز ثقل بدست می‌آید، جهت انحنا، میزان حرکت نسبی در جهت افقی و عمودی و نسبت طول به عرض مستطیل در برگیرنده قطعه [۱۹].

سپس این ویژگی‌ها با استفاده از مجموعه‌ای از قواعد، فازی شده‌اند که برای تشخیص بدنه حرف استفاده می‌شوند، همچنین نشانه حروف هم به عنوان یک ویژگی اضافی در تشخیص حرف استفاده می‌شود. در نهایت برای طبقه‌بندی از یک شبکه FLVQ استفاده شده است و نتیجه نهایی شناسایی ۹۰/۲۷ درصد به همراه ۳ درصد وازدگی گزارش شده است [۱۹].

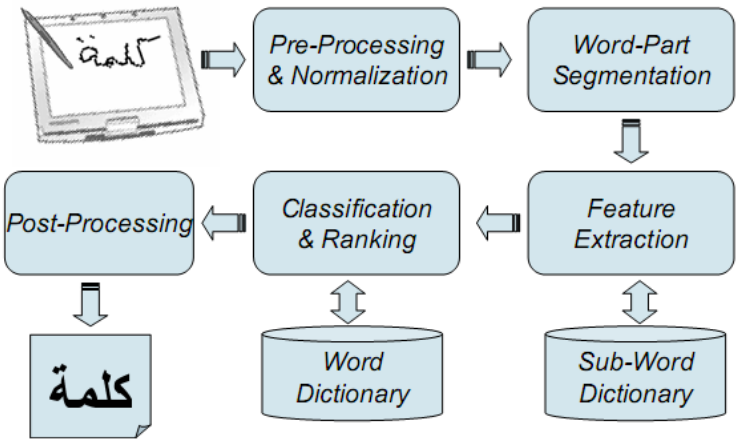
رضوی و کبیر [۲۰] حروف جدای فارسی را بر اساس علامت به ۱۲ گروه تقسیم کرده‌اند. سپس برای تشخیص یک حرف، ابتدا لازم است علامت آن شناسایی شود تا به یکی از این ۱۲ گروه نسبت داده شود، سپس بدنه حروف وارد شده فقط با حروف آن گروه مقایسه می‌شود. برای تشخیص نشانه، ابتدا محل نشانه از نظر بالا یا پایین بودن حروف بررسی می‌شود، سپس برای شناسایی نشانه از شبکه عصبی استفاده می‌شود. برای تشخیص بدنه حروف

مجموع فاصله اقلیدسی نقاط حرف نرمالیزه شده با نقاط حرف‌های نمونه گروه مقایسه می‌شود و ورودی به کلاس حرفی که کمترین فاصله را دارد نسبت داده می‌شود. درصد عملکرد این روش ۹۳/۳درصد گزارش شده است.

فرکی و پالهنگ [۲۱] در روش پیشنهادی خود ابتدا حروف جدای فارسی را بر اساس تعداد بخش‌ها به ۴ گروه تقسیم کرده‌اند. در این روش برای ساده‌تر شدن شناسایی ، پس از مشخص شدن گروه حرف، بر اساس نشانه‌های شناسایی‌شده‌ی حرف اقدام به کم‌کردن حروف کاندید می‌شود. سپس ویژگی‌هایی مانند نقاط مهم دستنوشته که با الگوریتم داگلاس و پکر[1] [۲۲] محاسبه می‌شوند و نقاط قله و دره استخراج شده و با استفاده از از یک طبقه‌بند مدل مخفی مارکوف با الگوریتم آموزش باوم-ولچ[2] اصلاح شده شناسایی حروف جداگانه انجام می‌شود. برای بررسی عملکرد روش پیشنهادی از ۵۳۹ حرف جداگانه فارسی نوشته شده توسط ۹ نفر برای آموزش و ۴۳۱ حرف نوشته شده توسط ۸ نفر برای تست استفاده شده است. برای این روش ۵/۱ درصد خطای مرحله آزمون اعلام شده است.

۲-۴- روش‌های تشخیص زیرکلمه جزئی نگر

Biadsy و همکاران [۲۳] سیستمی برای شناسایی کلمات عربی ارائه داده اند که از مدل مخفی مارکوف به همراه فرهنگ لغت زیرکلمات و مدل‌های شکل حروف برای شناسایی زیرکلمات استفاده می‌کند. اجزای اصلی این سیستم در شکل ۲-۵ نمایش داده شده است.

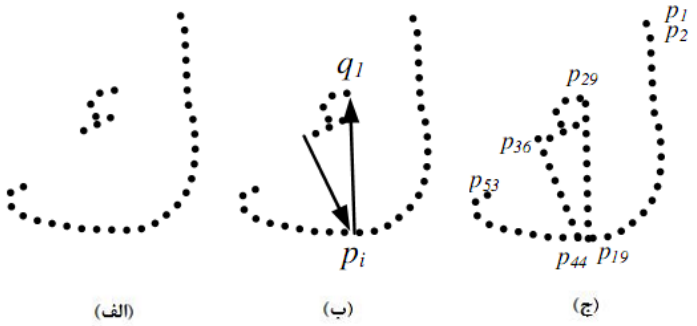


شکل ۲-۵: اجزای اصلی سیستم پیشنهادی در [۲۳]

[1] Douglas and Peucker
[2] Baum-Welch

در این روش ابتدا زیرکلمه با استفاده از الگوریتم داگلاس و پکر [۲۲] به قطعات کوچک تقسیم می‌شود. از هر قطعه سه ویژگی استخراج می شود، ویژگی اول زاویه بردار اتصال دهنده بین نقاط متوالی و ویژگی دوم زاویه بردار اتصال دهنده بین نقاط باقی مانده بعد از اعمال الگوریتم داگلاس و پکر (با حد آستانه کمتر از مرحله قطعه‌بندی) با محور افقی است و ویژگی سوم وجود یا عدم وجود حلقه در قطعه است که یک مقدار دودویی است.

به منظور تعامل با نشانه‌ها در این روش نشانه‌ها، با فرض این محدودیت که نشانه‌های زیرکلمه بلافاصله بعد از نوشتن آن گذاشته می‌شود، در مکان مناسبی از بدنه زیرکلمه جاسازی می‌شوند. این کار با استفاده از الگوریتم افکنش نشانه‌ها که شامل دو مرحله ، تعیین نشانه‌ها و الحاق آن‌ها به بدنه زیرکلمه در محل مناسب در رشته نقاط زیرکلمه، است، انجام شده است. برای این کار نقطه اول نشانه (q_1)، به نقطه بدنه زیرکلمه که مختصات عمودی نزدیک‌تری دارد (p_i) وصل می‌شود، آخرین نقطه حرکت نیز به نقطه بعدی بدنه زیرکلمه (p_{i+1}) وصل می‌شود. بین نقاط بدنه و حرکت، نقاط جدید اضافه می‌شوند تا بدنه و نشانه‌های آن با هم به یک حرکت تبدیل شوند. نمونه‌ای از این عملیات در شکل ۲-۶ نشان داده شده است [۲۳].



شکل ۲- ۶: (الف) زیرکلمه با یک نشانه (ب) اتصال نشانه به بدنه (ج) افزودن نقاط متصل کننده [۲۳].

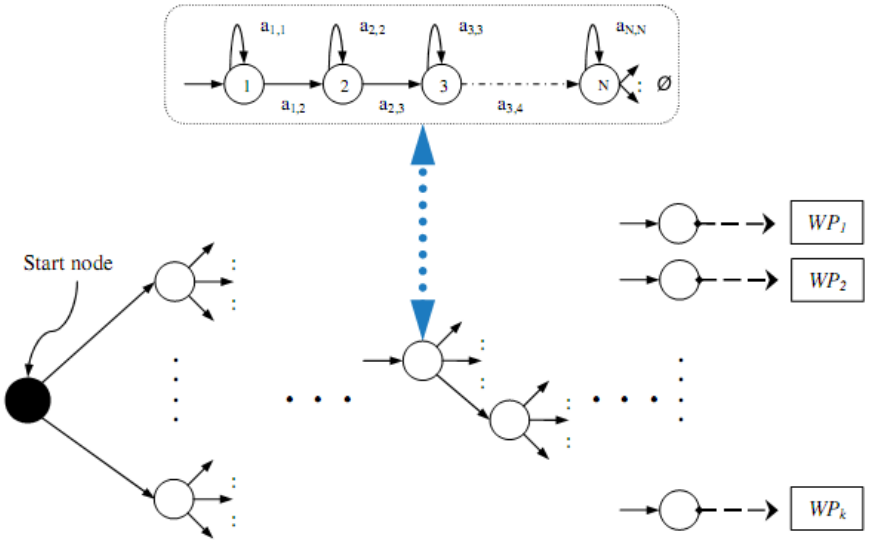
ویژگی‌های بکار رفته در این سیستم حاوی سه ویژگی هندسی زاویه محلی، قطعه بزرگ و حضور حلقه است که از رشته نقاط استخراج می‌شوند.

با اتصال علامت‌ها به بدنه زیرکلمه، شکل جدید حروف برای آموزش مدل مخفی مارکوف استفاده می‌شوند. بردار مشاهده مدل مخفی مارکوف باید دارای مقادیر صحیح مثبت باشد، به همین دلیل یک مرحله گسسته‌سازی با استفاده از کدهای فریمن ۱۶ و ۸ جهتی برای ویژگی‌های اول و دوم انجام شده است. این جهت ها و ویژگی دودویی سوم ۲۵۶ مقدار گسسته و وجود حرکت و موقعیت آن نسبت به بدنه زیرکلمه هم ۴ مقدار گسسته دیگر دارند که

در مجموع ۲۶۰ مشاهده مختلف را تشکیل می‌دهند. مدل مخفی مارکوف دارای توپولوژی چپ به راست است و تعداد حالت‌ها برای هر حرف بسته به میزان پیچیدگی هندسی آن بین ۵ تا ۱۱ انتخاب شده است.

با استفاده از فرهنگ لغت متنی، دو مجموعه فرهنگ کلمات D_i و فرهنگ زیرکلمات $WPD_{i,j}$ تشکیل شده است. هر مجموعه D_i شامل تمام کلمات دارای i زیرکلمه است و هر مجموعه $WPD_{i,j}$ شامل زیرکلمات j ام D_i است. برای تشخیص کلمه ورودی با k زیرکلمه، زیرکلمه n ام، فقط با زیرکلمات مجموعه $WPD_{k,n}$ مقایسه می‌شود.

تشخیص زیرکلمه با استفاده از مدل حروف آن انجام می‌شود، بدین ترتیب که برای هر زیرکلمه، یک شبکه از مدل‌های مخفی مارکوف حروف آن زیرکلمه تشکیل می‌شود. این شبکه در شکل ۲-۷ نشان داده شده است.

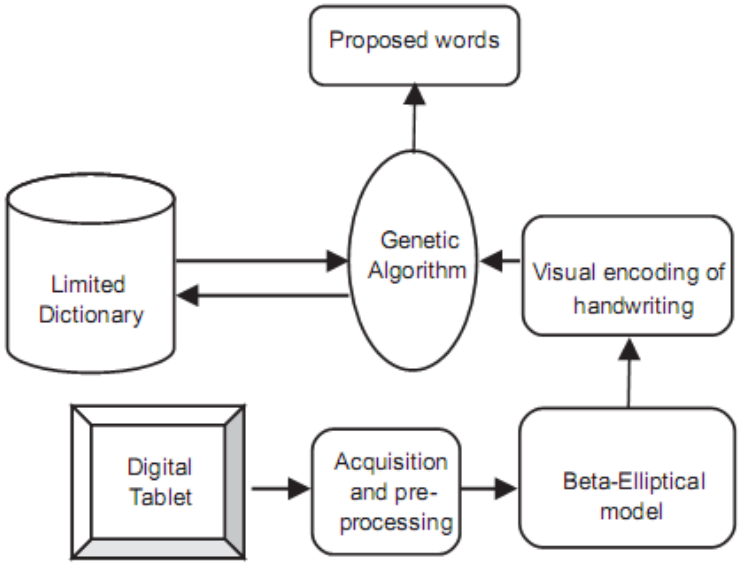


شکل ۲- ۷: مجموعه شبکه‌های زیرکلمات، هر گره در شبکه، یک مدل مخفی مارکوف مربوط به حرف زیرکلمه است و اتصالات بین گروه‌ها دارای وزن نیستند [۲۳].

زیرکلمه‌ای که بردار مشاهده زیرکلمه ورودی، خروجی شبکه مربوط به آن را بیشینه کند انتخاب می‌شود. بردار ویژگی زیرکلمه از اولین مشاهده تا زمانی که مدل اولین حرف بیشترین احتمال را تولید کند به عنوان بردار مشاهده اولین حرف زیرکلمه درنظر گرفته می‌شود، از ادامه بردار مشاهده تا زمانی که مدل حرف بعد در شبکه زیرکلمه، بیشترین احتمال را تولید کند به عنوان بردار مشاهده دومین حرف در نظر گرفته می‌شود و این روال تا پایان بردار مشاهده ادامه پیدا می‌کند. احتمال تولید زیرکلمه برابر با حاصل ضرب احتمال خروجی مدل حروف آن تعریف می‌شود و کلمات مجموعه D_k با حاصل ضرب خروجی شبکه‌های زیرکلمات آن‌ها امتیازبندی می‌شوند.

برای بررسی عملکرد سیستم ارائه شده از ۳۲۰۰ کلمه برای آموزش و ۲۳۵۸ کلمه از ۱۰ نویسنده برای تست استفاده شده است. نتیجه شناسایی روش با فرهنگ لغت با اندازه ۵۰۰۰، برابر۹۸/۴۴ درصد و برای فرهنگ لغت با اندازه ۴۰۰۰۰، ۹۵/۴۴ درصد گزارش شده است.

Kherallah و همکاران [۲۴] رویکرد خود را بر مبنای روانشناسی درک بصری دستنوشته قرار داده‌اند تا بتوانند توصیف اصیلی از دستنوشته داشته باشند. این سیستم در شکل ۲-۸ نشان داده شده است.



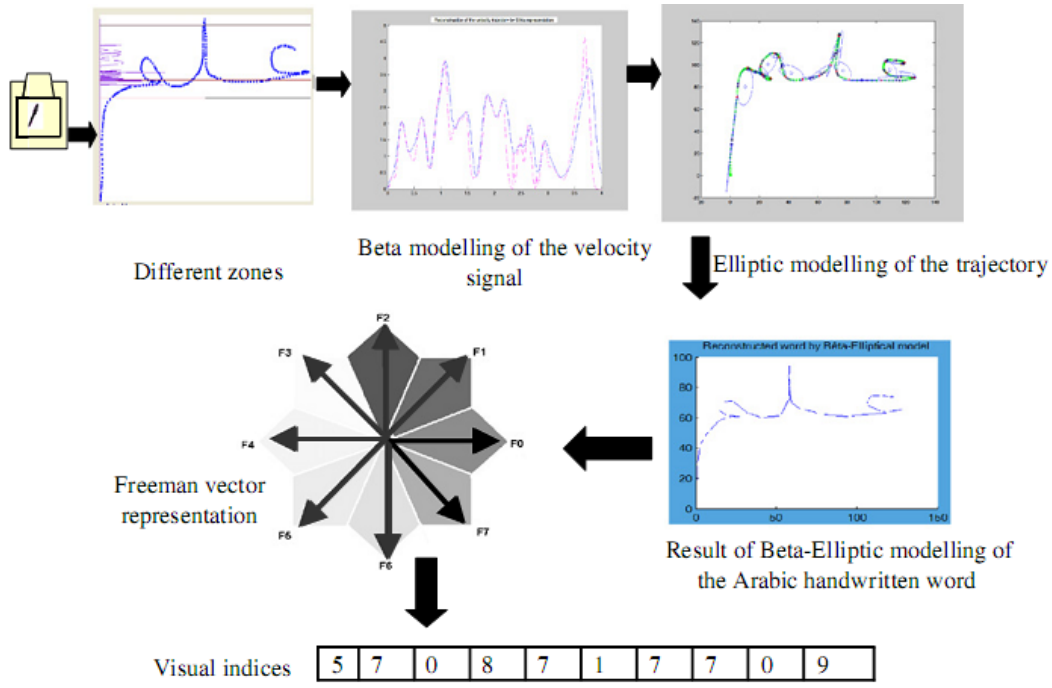
شکل ۲-۸: اجزای سیستم پیشنهادی در [۲۴]

آن‌ها در روش پیشنهادی خود پس از انجام پیش‌پردازش‌های لازم، در اولین گام برای استخراج ویژگی برمبنای مطالعات انجام شده بر روی حرکات پدیدآورنده دستنوشته، از مدل بتا- الیپتیکال [1] به عنوان شیوه‌ای متفاوت برای بازنمایی دستنوشته استفاده نموده‌اند. برای بدست آوردن مدل بیتا-الیپتیکال از نقاط بحرانی استفاده می‌شود، که در واقع این نقاط بیشینه‌های محلی در سیگنال سرعت دستنوشته هستند. در مرحله بعد مدل بتا-الیپتیکال نمایش دستنوشته به کدهای بصری تبدیل می‌شود. این کار نیز با استفاده از کد فریمن و تقسیم دستنوشه به چند ناحیه مختلف (ناحیه میانی و نواحی ثانویه) صورت می گیرد. در شکل ۲-۹ خلاصه ای از گام استخراج ویژگی نشان داده شده است. در گام شناسایی از الگوریتم ژنتیک استفاده شده است که یکی از دلایل بکارگیری این الگوریتم طول متغییر بردار ویژگی بوده است. هر عضو جمعیت این الگوریتم یک کلمه است که با کدهای بصری نمایش داده

[1] Beta-Elliptical

می‌شود، جمعیت اولیه الگوریتم از کدهای بصری ۲۰۰ کلمه به عنوان داده اولیه از ۵۰۰ کلمه جمع آوری شده توسط ۲۴ نویسنده مختلف انتخاب می‌شود. سپس با انجام دو عملیات برش[1] و جهش[2] که بر روی کدهای بصری انجام می‌شود، جمعیت جدید تولید می‌شود. در این الگوریتم تابع برازندگی[3] بر اساس شباهت هر عضو با کلمه ناشناخته ورودی کار می‌کند. بنابراین انتظار می‌رود تا در هر جمعیت جدید کلماتی شبیه تر به کلمه ناشناخته تولید شوند.

نتیجه شناسایی این روش بر روی ۵۰۰ کلمه عربی ۹۷ درصد گزارش شده است [۲۴].



شکل ۲-۹: نحوه استخراج ویژگی در سیستم [۲۴]

Al-Habian و Assaleh [۲۵] زیرکلمه عربی را به حرکت هایی که سازنده حروف عربی در مکان های مختلف زیرکلمه هستند تقسیم کرده‌اند که به عنوان واحدهای شناسایی استفاده می‌شوند، از ویژگی های Δx و Δy و sin و cos زاویه بین نقاط متوالی حرکت برای آموزش مدل مخفی مارکوف و شناسایی این حرکت‌ها استفاده شده است.

[1] Crossover
[2] Mutation
[3] Fittnes function

برای شناسایی این حرکت‌ها یک پنجره به طول ۱۰ نمونه روی ویژگی‌های استخراج شده حرکت می‌کند که دارای ۵۰ درصد همپوشانی است و مجموعه ویژگی‌ها را به طبقه‌بند مدل مخفی مارکوف می‌دهد. در بین ترکیب‌های مختلف، حالتی که بالاترین مقدار را داشته باشد انتخاب می‌شود. سپس با استفاده از درخت تصمیم گیری ساده، حالت‌های غیر معتبر حذف می شوند (مثل وجود حرکتی در ابتدای زیرکلمه که در ابتدای حروف اول وجود ندارد) سپس درصورتی که شکل نامعتبری از یک حرف در زیرکلمه پیدا شود، با نادیده گرفتن این شکل اشتباه، الگوریتم یک بار دیگر برای پیدا کردن شکل درست تکرار می‌شود. نتیجه نهایی ۷۸/۲۵ درصد تشخیص صحیح و ۳۴ درصد درج[1] برای حرکت‌ها و ۷۵/۲۵ درصد تشخیص صحیح و ۳۵/۷۵ درصد درج برای حروف گزارش شده است. منظور از درج، حرکت‌ها یا حروفی است که در نوشته وجود نداشته است اما سیستم تشخیص داده است.

حلاوتی و باغشاه [۲۶] با استفاده از ماشین کشسان فازی[2] (فم)، روشی برای تشخیص دستنوشته برخط فارسی ارائه کرده اند که بر خلاف مدل مخفی مارکوف نیاز به پیش فرض های احتمالاتی ندارند و با وجود دقت بازشناسی تقریبا مساوی، مقاومت بیشتری در مقابل تغییرات محیطی از خود نشان می‌دهد و سرعت مناسبی نیز دارد.

با استفاده از زاویه بردارهای بین نقاط متوالی، هر حرکت به قسمت‌هایی تقسیم شده است که سه نوع کلی خط، کمان و نیم حلقه هستند. با توجه به فاصله چند نقطه اول از نوشته کاربر، واحدی برای طول مشخص می‌شود که بر اساس این واحد به هر قطعه یک طول فازی نسبت داده می‌شود. یک ویژگی فازی دیگر، جهت حرکت است که از زاویه خط بین نقاط ابتدایی و انتهایی بدست می‌آید و دارای ۸ مقدار است. ویژگی غیرفازی انحنا هم برای کمان‌ها و نیم حلقه‌ها تعریف می‌شود به این صورت که اگر نقطه ثقل در سمت چپ نقطه شروع قرار داشته باشد، ساعت گرد و در غیر این صورت پادساعت گرد است.

پایگاه داده بکار رفته شامل ۱۲۵۰ کلمه از کتاب‌های اول دبستان نوشته شده توسط ۲۰ نفر است و نتیجه نهایی ۷۸ درصد بدون فرهنگ لغت و ۹۶ درصد با فرهنگ لغت است [۲۷].

[1] Insertion
[2] Fuzzy Elastic Machine (FEM)

فصل ۳

تئوری های مرتبط بکار رفته

فصل ۳- تئوری های مرتبط با کار انجام شده

۳-۱- انطباق الگو[1]

ماهیت مسئله ای که در مبحث انطباق الگو با آن روبرو هستیم اندکی با دیگر حوزه‌های شناسایی الگو متفاوت است. در مبحث انطباق الگو فرض ما بر این است که مجموعه‌ای از الگوهای مرجع موجودند و هدف ما آن است که تصمیم بگیریم که کدام یک از الگوهای مرجع بیشترین شباهت را با یک الگو تست ناشناخته دارد، درحالی که در دیگر حوزه ها عمده تمرکز ما بر روی آن است تا یک الگو ناشناخته را به یکی از چند کلاس ممکن اختصاص دهیم [۲۸].

انطباق الگو در زمینه های مختلفی کاربرد دارد، مثل پردازش زبان طبیعی جایی که از فاصله اصلاح برای تصحیح املای کلمات استفاده می‌شود. بدین شکل که کلماتی از فرهنگ لغت که با کلمه با املای اشتباه شباهت بیشتری دارند، به عنوان کلمات کاندید برای جایگزینی انتخاب می شوند. یا در بیوانفورماتیک برای اندازه گیری شباهت دو ماکرومولکول مثل DNA که معمولاً بصورت رشته هایی از حروف نمایش داده می‌شوند از انطباق الگو استفاده می‌شود.

در اینجا شاید این مسئله مطرح شود که اگر هر الگو بشکل بردار یا ماتریسی از ویژگی ها نمایش داده می‌شود، چرا از معیارهای اندازه‌گیری مثل فاصله اقلیدسی یا نرم‌های فروبنیوس[2] استفاده نمی‌کنیم؟ با کمی تفکر بیشتر مشخص می شود که چنین رویکردهای سرراستی کافی نیستند و به چیز بیشتری نیاز است. بطور مثال ممکن است طول الگوهای مرجع و تست به هیچ وجه هم اندازه نباشند [۲۸].

برای درک بهتر این موضوع، یک مسئله انطباق متن دستنویس را درنظر بگیرید که قرار است تعیین کنیم که کلمه نوشته شده، کدام یک از کلمات مجموعه شناخته شده ماست. مثلاً "beauty" را درنظر بگیرید که ممکن است بدلیل خطای سنسورها شبیه به "beety" یا "beaut" ظاهر شود که طول متفاوتی نسبت به کلمه اصلی دارند.

در اولین گام برای مبحث انطباق الگو منطقی است تا به تعریف یک معیار یا هزینه برای اندازه گیری "فاصله" یا "شباهت" بین الگوهای مرجع (که شناخته شده اند) با الگوی تست (که ناشناخته است) بپردازیم تا بتوانیم عملیات انطباق این دو را که به عنوان تطابق الگو می‌شناسیم انجام دهیم.

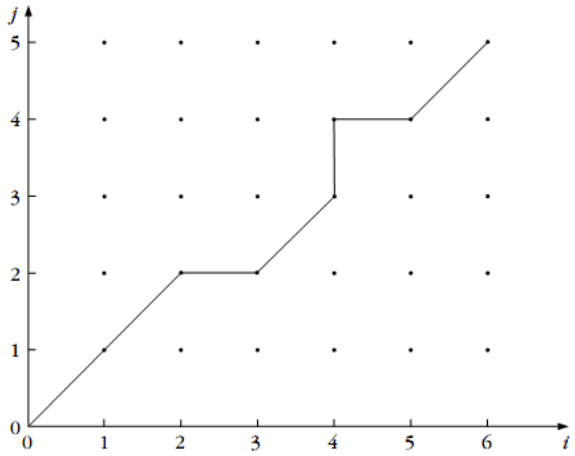
---

[1] Template matching

[2] Frobenius norms

۳-۱-۱- معیارهای اندازه گیری و جستجوی مسیر بهینه

همانطور که قبلاً ذکر شد به عنوان اولین گام در مبحث انطباق الگو منطقی است تا به تعریف یک معیار یا هزینه برای اندازه‌گیری "فاصله" یا "شباهت" بین الگوهای مرجع با الگوی تست بپردازیم. ما در اینجا بر روی انطباق الگوهایی تمرکز می‌کنیم که شامل رشته‌ای از سمبل های مشخص هستند. فرض کنید که داشته باشیم $r(i), i = 1,2, \dots, I$ و $t(j), i = 1,2, \dots, J$ که بردار ویژگی متناظر با یک جفت الگوی مرجع و تست باشند، و بطور کلی $I \neq J$ است. هدف آن است تا معیار اندازه‌گیری فاصله مناسبی بین دو رشته ارائه شود. برای این کار ما یک شبکه دو بعدی در نظر می‌گیریم که عناصر دو رشته بر روی دو محور آن قرار می‌گیرند. رشته مرجع بر روی محور افقی و رشته تست بر روی محور عمودی قرار می‌گیرد. شکل ۳-۱ مثالی برای $I = 6$ و $J = 5$ را نشان می‌دهد.



شکل ۳-۱: هر نقطه از مسیر بیانگر تشابه بین عناصر متناظر از رشته مرجع و تست است [۲۸].

هر نقطه از شبکه (گره) تشابه بین عناصر متناظر از هر دو رشته را نشان می‌دهد. برای مثال گره $(3,2)$ عنصر $r(3)$ را به $t(2)$ نگاشت می کند. هر گره (i,j) از شبکه با یک هزینه در ارتباط است که توسط تابع $d(i,j)$ به عنوان فاصله بین دو عنصر متناظر $t(j)$ و $r(i)$ اندازه گیری می‌شود.

یک مسیر از شبکه از گره شروع (i_0, j_0) به گره پایانی (i_f, j_f) از یک مجموعه گره‌ها به فرم زیر تشکیل می‌شود.

$$(i_0, j_0), (i_1, j_1) , (i_2, j_2), \dots, (i_f, j_f) \qquad (۱-۳)$$

هر مسیر با یک هزینه کلی D مرتبط است که برابر است با:

$$D = \sum_{k=0}^{K-1} d(i_k, j_k) \qquad (۲-۳)$$

طوریکه K تعداد گره های در مسیر است. برای مثال در شکل ۳-۱، $K = 8$ است.

هزینه کلی تا گره (i_k, j_k) با $D(i_k, j_k)$ نشان داده می شود و فرض می کنیم که $D(0,0) = 0$ و همچنین $d(0,0) = 0$ باشد. همچنین یک مسیر را کامل گوئیم اگر:

$$(i_0, j_0) = (0,0), (i_f, j_f) = (I, J) \qquad (۳-۳)$$

در این حالت کمترین D بر روی تمام مسیرهای ممکن به عنوان فاصله بین دو رشته تعریف می شود. در این جا قبل از آنکه وارد بحث بهینه‌سازی شویم بایستی به تغییراتی که می‌توان در این طرح کلی داد اشاره کنیم. بطور مثال می‌توان به هر گذار بین دو گره هزینه‌ای اختصاص داد و هزینه بعضی از گذارهای خاص را بیشتر درنظر گرفت. در این حالت، هزینه در گره (i_k, j_k) به گذاری که به این گره صورت گرفته نیز بستگی داشته بدین شکل که از کدام گره (i_{k-1}, j_{k-1}) به گره (i_k, j_k) دسترسی پیدا شده است. بنابراین هزینه d به فرم $d(i_k, j_k | i_{k-1}, j_{k-1})$ خواهد بود و هزینه کلی برابر است با:

$$D = \sum d(i_k, j_k | i_{k-1}, j_{k-1}) \qquad (۴-۳)$$

اجازه دهید به مسئله بهینه سازی برگردیم. برای تعیین بهترین مسیر، یک راه حل آن است تا تمام ترکیب مسیرها بررسی شوند. اما این کار فرایند محاسباتی پر هزینه ای دارد. الگوریتم های برنامه نویسی پویا که بر مبنای اصل بلمن هستند ابزارهای قدرتمندی برای کاهش پیچیدگی محاسباتی هستند.

۳-۱-۲- اصل بهینگی بلمن[1] و برنامه نویسی پویا

اجازه دهید مسیر بهینه بین گره اولیه (i_0, j_0) و گره پایانی (i_f, j_f) به شکل زیر نمایش داده شود.

$$(i_0, j_0) \xrightarrow{opt} (i_f, j_f) \qquad (۵-۳)$$

اگر (i, j) یک گره میانی بین (i_0, j_0) و (i_f, j_f) باشد، ما مسیر بهینه با قید عبور از (i, j) را بدین شکل نشان خواهیم داد.

[1] Belman Optimality

$$(i_0, j_0) \underset{(i,j)}{\overset{opt}{\longrightarrow}} (i_f, j_f) \qquad (۶-۳)$$

اصل بلمن بیان می دارد که:

$$(i_0, j_0) \underset{(i,j)}{\overset{opt}{\longrightarrow}} (i_f, j_f) = (i_0, j_0) \overset{opt}{\longrightarrow} (i,j) \oplus (i,j) \overset{opt}{\longrightarrow} (i_f, j_f) \qquad (۷-۳)$$

جاییکه $\oplus$ الحاق مسیرها را نشان می دهد. به عبارت دیگر، اصل بلمن بیان می دارد که مسیر بهینه از (i_0, j_0) به (i_f, j_f) با عبور از (i,j) الحاقی از مسیر بهینه از (i_0, j_0) به (i,j) و مسیر بهینه از (i,j) به (i_f, j_f) است. نتیجه این اصل آن است که وقتی ما در (i,j) بر روی مسیر بهینه هستیم، آنگاه برای دستیابی بهینه به (i_f, j_f) نیاز داریم تا تنها به دنبال مسیر بهینه از (i,j) به (i_f, j_f) بگردیم.

اجازه دهید این مسئله را طوری بیان کنیم که بعداً برای ما مفید باشد. فرض کنید که ما از (i_0, j_0) شروع کرده و گره kام از مسیر (i_k, j_k) باشد. هدف محاسبه کمترین هزینه لازم برای دستیابی به گره آخر است. گذار به (i_k, j_k) از یکی از گره های ممکنی انجام می‌شود که می‌توانند در مکان $k-1$ ام از مسیر قرار گیرند. این موضوع مهم است که برای هر گره از شبکه ما فرض کنیم که مجموعه‌ای از گره‌های پیشین تعریف شده است، که آن را قید محلی نیز گویند. اصل بلمن موجب می شود تا:

$$D_{min}(i_k, j_k) = \min_{i_{k-1}, j_{k-1}} [D_{min}(i_{k-1}, j_{k-1}) + \mathrm{d}(i_k, j_k | i_{k-1}, j_{k-1})] \qquad (۸-۳)$$

بنابراین، هزینه کمینه کل برای دسترسی به گره (i_k, j_k) برابر است با هزینه کمینه گره (i_{k-1}, j_{k-1}) به اضافه هزینه اضافی گذار از (i_{k-1}, j_{k-1}) به (i_k, j_k). پس جستجو برای هزینه کمینه تنها به مجموعه‌ای از گره‌های پیشین (i_k, j_k) مرتبط می‌شود. این رویه برای تمام گره‌های شبکه انجام می‌شود. الگوریتم حاصل از این رویه به عنوان برنامه نویسی پویا[1] شناخته می‌شود.

[1] Dynamic programming

۳-۱-۳- فاصله اصلاح[1]

در علوم کامپیوتر فاصله اصلاح راهی برای بیان کمّی تفاوت دو الگو است که با شمارش کمترین تعداد عملیات لازم برای تبدیل یک رشته به دیگری محاسبه می‌شود.

در این اینجا بدلیل اینکه الگوهای ما رشته‌هایی حاویی مجموعه‌ای از سمبل‌های مشخص هستند در نتیجه با فرض اینکه دو رشته t و r بر روی الفبای $\sum$ (مثلاً مجموعه کاراکترهای ASCII یا کدهای فریمن) داشته باشیم، فاصله اصلاح $D(t,r)$ کمترین وزن یک سری از عملیاتهای تعریف شده روی رشته است که باعث تبدل رشته t به r شود.

۳-۱-۴- فاصله لونشتین[2]

بر مبنای مجموعه عملیاتهایی که بر روی رشته می‌توان انجام داد چند تعریف مختلف از فاصله اصلاح وجود دارد. یکی از معمول‌ترین این تعاریف فاصله لونشتین است. در تعریف لونشتین، عملیات‌های روی رشته عبارتند از:

- درج کردن[3] یک سمبل. اگر $a = uv$ باشد، درج کردن سمبل x باعث تولید uxv خواهد شد.
- حذف[4] یک سمبل که $a = uxv$ را به uv تغییر خواهد داد.
- جایگزینی[5] یک سمبل مثل x با سمبلی مثل $y \neq x$ که باعث تبدیل uxv به uyv خواهد شد.

در تعریف اصلی لونشتین هر یک از این عملیات‌ها دارای هزینه واحد هستند، بنابراین فاصله لونشتین برابر مینیمم تعداد عملیات‌های لازم برای تبدیل t به r است.

واضح است که ترکیبی از این عملیات‌ها نیز ممکن است اتفاق افتد. اگر رشته‌ها طول یکسانی داشته باشند آنگاه هزینه مستقیماً به تعداد سمبل‌هایی وابسته است که از یک رشته بایستی تغییر کنند تا رشته دیگر نتیجه شود. قضیه وقتی جالبتر می شود که طول دو رشته یکسان نباشد. در این حالت سمبل ها بایستی در مکان مشخصی از رشته تست حذف و یا اضافه شوند. فاصله اصلاح بین دو رشته t و r که با $D(t,r)$ نشان داده می‌شود، به صورت کمترین تعداد جایگزینی C، اضافه کردن I، و حذف کردن R لازم برای تبدیل الگوی t به الگوی r تعریف می شود.

---

[1] Edit distance
[2] Levenshtein distance
[3] Insert
[4] Deletion
[5] Substitution

$$D(a,b) = min_j[C(j) + I(j) + R(j)] \quad (۹-۳)$$

جائیکه j تمام ترکیبات ممکن از تغییرات سمبل‌ها برای بدست آمدن r از t را نشان می‌دهد. برای شرح دقیقتر مسئله، توجه داشته باشید که بطور مثال بیش از یک راه برای تغییر "beuty " به "beauty " وجود دارد. برای مثال می‌توان "a" را بعد از "e" اضافه کرد و یا اینکه "u" را به "a" تغییر داده و بعداً "u" را اضافه کرد.

در تعریف کلی‌تر، برای هر عملیات روی رشته یک تابع وزن w درنظر گرفته می‌شود. در این حالت با استفاده از عملیات‌های لونشتین فاصله اصلاح بین دو رشته t و r با رابطه بازگشتی زیر تعریف می شود:

$$D(i,0) = \sum_{k=1}^{i} w_{ins}(r_k), \ \ 1 \le i \le I \quad (۱۰-۳)$$

$$D(0,j) = \sum_{k=1}^{j} w_{del}(t_k), \ \ 1 \le j \le J \quad (۱۱-۳)$$

$$D(i,j) = \begin{cases} D(i-1,j-1) & t_i = r_j \\ min \begin{cases} D(i-1,j) + w_{ins}(r_j) \\ D(i,j-1) + w_{del}(t_i) \\ D(i-1,j-1) + w_{sub}(t_i, r_j) \end{cases} & t_i \neq r_j \quad 1 \le i \le I \ \ ,1 \le j \le J \end{cases} \quad (۱۲-۳)$$

استفاده از رابطه بازگشتی بالا برای ارزیابی فاصله اصلاح دارای پیچیدگی زمانی نمایی است. به همین دلیل معمولاً از تکنیک برنامه نویسی پویا برای محاسبه فاصله اصلاح استفاده می شود. در این حالت پیچیدگی زمانی به $O(IJ)$ کاهش می‌یابد.

۳-۱-۵- جستجوی مسیر بهینه با استفاده از تکنیک برنامه نویسی پویا

در اینجا تکنیک برنامه نویسی پویا برای حل مسئله بدون در نظر گرفتن توابع وزن شرح داده می‌شود. برای این کار ما با قرار دادن سمبل‌ها رشته مرجع بر روی محور افقی و سمبل‌های رشته تست بر روی محور عمودی یک شبکه تشکیل می‌دهیم. اولین گام در فرایند جستجوی مسیر بهینه با استفاده از تکنیک برنامه نویسی پویا آن است تا قیود گذار از گره‌های شبکه شرح داده شود. در مسئله پیش روی ما این قیود عبارتند از:

هزینه D(0,0) از گره (0,0) برابر صفر است.

جستجو برای یک مسیر کامل انجام شود.

هر گره (i,j) می تواند تنها از سه گره پیشین خود قابل دسترس باشد که در زیر آمده اند.

$$(\mathrm{i}-1,\mathrm{j}),(\mathrm{i}-1,\mathrm{j}-1),(\mathrm{I},\mathrm{j}-1) \qquad (۱۳-۳)$$

هزینه متناظر با سه گذار بالا عبارتند از:

- گذار قطری:

$$d(i,j|i-1,j-1)=\begin{cases}0 & if\ r(j)=t(j)\\ 1 & if\ r(j)\neq t(j)\end{cases}$$

یعنی هزینه یک گذار برابر صفر است اگر سمبل های متناظر با گره (i,j) مشابه باشند و برابر یک است اگر آنها متفاوت باشند.

- گذار افقی و عمودی:

$$d(i,j|i-1,j)=d(i,j|i,j-1)=1$$

معنای گذارهای افقی آن است که مبادرت به هم ترازی دو رشته با درج یک سمبل صورت گرفته است. بنابر این یک واحد به هزینه اضافه می شود. بطور مشابه گذار عمودی یک واحد به هزینه اضافه می‌کند زیرا این کار دلالت بر انجام عمل حذف است.

با کنار هم قرار دادن قیود و فاصله در رویه برنامه نویسی پویا، الگوریتم زیر حاصل می شود [۲۹].

$$D(0,0)=0$$

$$For\ i=1\ to\ I$$

$$\quad D(i,0)=D(i-1,0)+1$$

$$End\{For\}$$

$$For\ j=1\ to\ J$$

$$\quad D(0,j)=D(0,j-1)+1$$

$$End\{For\}$$

$$For\ i=1\ to\ I$$

$$\quad For\ j=1\ to\ J$$

$$\quad\quad c1=D(i-1,j-1)+d(i,j|i-1,j-1)$$

$$\quad\quad c2=D(i-1,j)+1$$

$$\quad\quad c3=D(i,j-1)+1$$

$$D(i,j) = \min(c1, c2, c3)$$

$End\{For\}$

$End\{For\}$

$D(A,B) = D(I,J)$

به عبارت دیگر، ما ابتدا به محاسبه مینیمم هزینه برای دسترسی به هر گره از شبکه می پردازیم که از $(0,0)$ شروع شده و مسیر بهنیه کامل جز به جز محاسبه می‌شود.

درصورتی که برای هر عملیات تابع وزن w درنظر گرفته شود، الگوریتم بکار رفته شبیه به الگوریتم قبل است با این تفاوت که هزینه گذار دیگر برابر مقدار واحد نخواهد بود و بایستی تابع وزن w به جای هزینه گذار واحد در الگوریتم قبل اعمال شود.

فصل ۴

روش پیشنهادی

فصل ۴- روش پیشنهادی

۴-۱- پیشگفتار

در تحقیق انجام شده روشی برای تشخیص زیرکلمات برخط فارسی پیشنهاد شده است. روش پیشنهادی یک روش با رویکرد تجزیه‌ای بوده که در آن قطعه‌بندی آشکار وجود ندارد و مستقیماً به جستجوی حروف تشکیل‌دهنده زیرکلمه می‌پردازد.

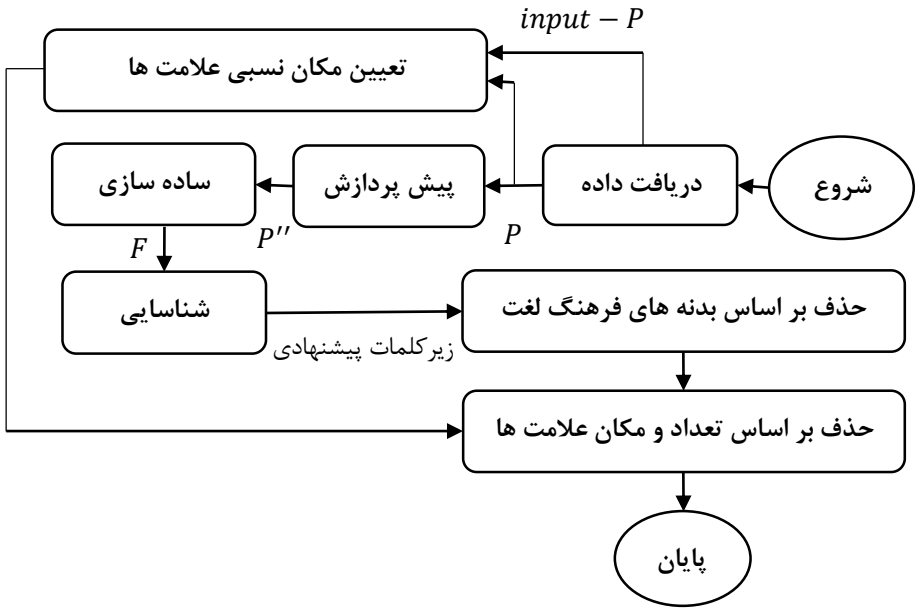
به دلیل قابلیت سیستم‌های سخت افزاری با صفحه لمسی به راحتی امکان جداسازی بدنه اصلی دستنوشته از دیگر حرکات آن وجود دارد، در نتیجه ورودی روش پیشنهادی ما مجموعه‌ای از حرکات قلم شامل بدنه اصلی و علامت‌ها خواهد بود که این حرکات قلم بصورت مجموعه‌ای از نقاط در صفحه می‌باشند.

بطور کلی شکل‌های نوشتاری مختلفی که برای علامت‌های حروف از جمله : کلاه "آ"، تک نقطه، دو نقطه، سه نقطه، همزه، سرکش‌های "ک/گ" و دسته "ط/ظ" بکار می‌رود، و از طرفی عدم رعایت نویسندگان در قرار دادن علامت‌ها در مکان مناسب ، باعث پیچیده شدن بیشتر تشخیص دستنوشته برخط می‌شود. به همین دلیل روش پیشنهادی در گام شناسایی تنها بر روی بدنه اصلی زیرکلمه متمرکز است و از علامت‌ها و تعداد آن‌ها در گام‌های بعدی به عنوان فیلتر استفاده می‌شود و گام پیش‌پردازش بر روی آن‌ها انجام نمی‌شود.

بنابراین ورودی مرحله بازشناسی بدنه اصلی زیرکلمه ورودی است که در ابتدا یک سری پیش‌پردازش‌های لازم بر روی آن صورت گرفته است. خروجی مرحله بازشناسی مجموعه‌ای از پیشنهادات به شکل رشته‌هایی از بدنه حروف خواهد بود. معمولاً تعداد زیادی از این پیشنهادات به شکلی هستند که هرچند امکان نوشتن آن‌ها وجود دارد اما در فرهنگ لغت فارسی وجود ندارند، به همین دلیل در گام پس پردازش می‌توان پیشنهاداتی که بدنه اصلی آن‌ها در فرهنگ لغت فارسی وجود نداشته باشد را حذف کرد. در تحقیق انجام شده از خود زیرکلمات پایگاه داده به عنوان فرهنگ لغت سیستم استفاده شده است.

قبلاً ذکر شد که استفاده از علامت‌ها تا حدودی مشکل است، اما از تعداد و محل قرارگیری آن‌ها می‌توان به عنوان فیلتری موثر جهت شناسایی استفاده نمود. نحوه انجام کار بدین صورت است که تعداد علامت‌ها و محل قرارگیری آن‌ها نسبت به بدنه اصلی مشخص می‌شود و از آن برای فیلترکردن پیشنهادات استفاده می‌شود.

بلوک دیاگرام کامل سیستم پیشنهادی در شکل ۴-۱ نشان داده شده است که در ادامه تمام قسمت‌های آن با جزئیات کامل توضیح داده خواهد شد.



شکل ۴-۱: بلوک دیاگرام کامل روش پیشنهادی

۴-۲- پیش پردازش

قبلاً به این موضوع اشاره کردیم که داده‌های موجود در پایگاه داده مورد استفاده به صورت رشته‌ای از نقاط در فایل ذخیره شده‌اند. در اولین گام برای آماده‌سازی پایگاه داده، ابتدا نقاط ورودی برای هر حرکت قلم را جدا نموده و به شکل زوج‌های (x,y) ذخیره می‌کنیم. برای سادگی کار مجموعه نقاط را به شکلی انتقال می‌دهیم که تمام مقادیر عددی مثبت شوند. بنابر این ورودی سیستم برای هر ورودی پایگاه داده، از یک یا چند حرکت قلم تشکیل شده که اولین حرکت مربوط به بدنه و مابقی مربوط به علامت‌ها خواهد بود. پس هر حرکت قلم از تعدادی نقاط در صفحه با مقادیر صحیح غیرمنفی تشکیل شده است که آن را با P نمایش می‌دهیم.

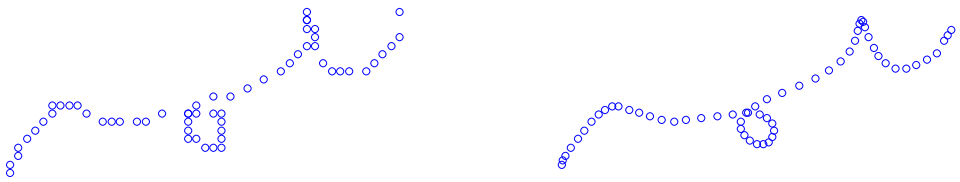
$$P = \{P_i = P(x_i, y_i)\} \ , i = 1 \dots n \qquad (۱-۴)$$

به دلیل این که کار شناسایی بر روی بدنه اصلی زیرکلمه متمرکز است، پیش‌پردازش تنها بر روی بدنه اصلی انجام می‌شود و از اعمال آن بر روی علامت‌ها صرف نظر می‌کنیم.

۴-۲-۱- هموارسازی

پس از تغییر فرمت داده‌های ورودی، برای حذف نویز مربوط به حرکت قلم ابتدا لازم است تا هموارسازی انجام شود. هرچند در مواردی ممکن است هموارسازی باعث از بین رفتن اطلاعات مهمی در دستنوشته شود، مانند دندانه‌های بسیار ریز در نوشته، ولی بدلیل نویز زیاد موجود هموارکردن نوشته کاری ضروری است.

در روش پیشنهادی برای انجام هموارسازی از یک فیلتر میانگین متحرک[1] استفاده شده است. نحوه انجام کار بدین صورت است که مختصات هر نقطه از بدنه اصلی در دو بعد افقی و عمودی به صورت جداگانه با میانگین مختصات چند نقطه مجاور خود جایگزین می‌شود. هر چه تعداد نقاط مورد استفاده در میانگین‌گیری بیشتر باشد، هموارسازی بیشتری صورت می‌گیرد. به همین دلیل بطور تجربی برای هموارسازی از دو نقطه قبل و بعد از هر نقطه برای میانگین‌گیری استفاده می‌کنیم تا اطلاعات کمی از دست داده شود. نمونه‌ای از انجام هموارسازی در شکل ۴-۲ نشان داده شده است.



شکل ۴-۲: (سمت چپ) نمونه ورودی. (سمت راست) بعد از هموارسازی

در این حالت تغییری در تعداد نقاط نمونه ورودی نداریم هرچند مقادیر مختصات به شکل اعشاری در خواهند آمد. لذا خواهیم داشت:

[1] Moving Average

$$P' = \{P'_i = P'(x_i, y_i)\} \ , i = 1 \dots n \qquad (۲-۴)$$

۴-۲-۲- نمونه برداری مجدد

فاصله هندسی بین نقاط زیرکلمه ورودی متناسب با سرعت حرکت قلم است و سرعت حرکت قلم برای نویسندگان مختلف متفاوت است. از طرفی هر شخص نیز ممکن است نوشته خود را در شرایط مختلف با سرعت‌های متفاوت بنویسد. در روش پیشنهادی به دلیل این که تنها از ویژگی‌های شکل نوشته استفاده می‌شود و بنا به ساده‌سازی که در ادامه به آن خواهیم پرداخت، استفاده از ویژگی سرعت دستنوشته برای ما کاربرد ندارد، بنابر این به وسیله نرمالسازی از نوع نمونه برداری مجدد سرعت حرکت قلم را حذف می‌کنیم. با نمونه برداری مجدد فاصله هندسی بین هر دو نقطه از نمونه برابر خواهد بود که ما این فاصله را بشکل تجربی برابر مقدار کوچک $D = 0.5$ در نظر گرفته‌ایم تا اطلاعات چندانی از دستنوشته از بین نرود.

برای نمونه برداری مجدد بدین شکل عمل می‌کنیم که اگر فاصله نقطه i ام از نقطه $i+1$ ام کمتر از D باشد، یک نقطه در فاصله D از نقطه i ام قرار داده و نقطه $i+1$ ام را حذف می‌کنیم و در صورتی که فاصله دو نقطه از D بیشتر باشد، تا وقتی به فاصله کمتر از D از نقطه $i+1$ ام برسیم نقاط جدید بر روی خط واصل دو نقطه قرار می‌دهیم.

با انجام نمونه برداری مجدد تعداد نقاط نمونه ورودی تغییر می‌کند، بنابر این داریم:

$$P'' = \{P''_i = P''(x_i, y_i)\} \ , i = 1 \dots m \qquad (۳-۴)$$

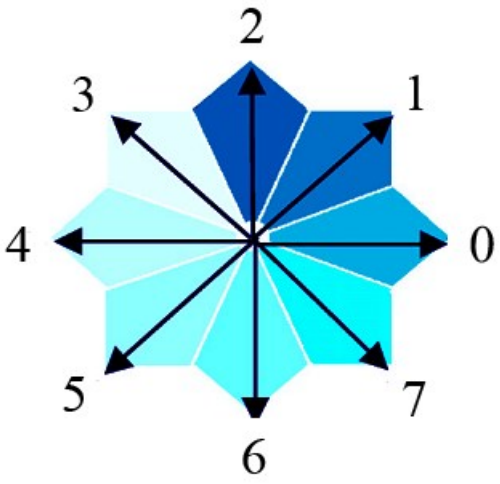
در شکل ۴-۳ نمونه ای از عملیات نمونه برداری مجدد نشان داده شده است.

شکل ۴-۳: (سمت چپ) نمونه هموار شده. (سمت راست) بعد از نمونه برداری مجدد

۴-۳- ساده سازی

۴-۳-۱- کد زنجیره ای

برای استخراج ویژگی، در گام اول رشته نقاط P'' بر اساس کد زنجیره ای ۸ جهته به مجموعه بردارهای V تبدیل می‌شود. برای انجام این کار زاویه مابین هر دو نقطه از رشته P'' محاسبه شده و به یکی از ۸ جهت تعریف شده در شکل ۴-۴ تبدیل می‌شود.



شکل ۴-۴: کد زنجیره ای هشت جهته

با این تبدیل به جای یک رشته m نقطه‌ای، رشته ای با طول $m-1$ از جهات که با مقادیر صفر تا ۷ نشان داده می‌شوند خواهیم داشت:

$$V = \{V_i\}, i = 1 \dots m-1, V_i \in \{0,1,2,3,4,5,6,7\} \quad (۴-۴)$$

۴-۳-۲- فشرده سازی

در گام بعدی برای سادگی بیشتر نحوه نمایش رشته جهات V را تغییر می دهیم. برای اینکار با فشرده‌سازی رشته جهتی V، آن را با رشته‌ای از زوج‌های (d, l) به نام F نمایش می‌دهیم که d بیانگر یک جهت و l بیانگر تعداد بردارهای در جهت مربوطه است به گونه‌ای که:

$$V = \{V_i\}, i = 1 \dots m-1 \ , F = \{(d_i, l_i)\} \ \ i = 1 \dots k \leq m-1$$
$$d_i \in \{0,1,2,3,4,5,6,7\}, l_i > 0 \ , \sum_{i=1}^{k} l_i = m-1 \quad (۵-۴)$$

به طور مثال برای رشته جهتی زیر:

$$V = \{2,1,,0,0,0,7,7,6,5,5,,4,4,4,4,4,5,5,5,6,6,6,6,6,6,6\}$$

شکل فشرده شده آن بصورت زیر خواهد بود:

$$F = \{(2,1),(1,1),(0,3),(7,2),(6,1),(5,2),(4,5),(5,3),(6,7)\}$$

با این کار در واقع عنصر دوم هر زوج در رشته F یعنی l_i متناسب با طول جهت متناظر خود در دستنوشته خواهد بود، زیرا از مرحله نمونه برداری مجدد به خاطر داریم که فاصله بین هر دو نقطه و در نتیجه اندازه تمام بردارهای جهتی یکسان هستند، بنابر این l_i بیانگر طول قسمتی از دستنوشته است که در جهت d_i است.

۴-۳-۳- حذف جهات قطری

برای سادگی بیشتر، هر جهت قطری را می‌توان به وسیله دو جهت اصلی بازسازی کرد. لذا در گام نهایی برای استخراج ویژگی اقدام به جایگزینی زوج‌های با جهات قطری با جهات اصلی می‌کنیم. این کار باعث می‌شود تا رشته زوج F که معرف دستنوشته ورودی است و در واقع ویژگی‌های مورد استفاده ما در مرحله شناسایی است تا حد امکان ساده شود. برای حذف جهات قطری اگر اندازه هر جفت قطری در F از حد آستانه‌ای بزرگتر باشد با دو جفت افقی و عمودی جایگزین می‌شود و در غیر اینصورت حذف می‌شود. برای جایگزینی، مجموعه نقاط P'' متناظر با جفت قطری استخراج شده و خط متصل‌کننده دو نقطه ابتدایی و انتهایی این زیر رشته از نقاط که آن را L می‌نامیم را بدست می‌آوریم. سپس نقاط متناظر با جفت قطری بر اساس آن که بالا یا پایین خط L قرار می‌گیرند به دو دسته Up یا $Down$ تقسیم بندی می‌شوند.

حال بر اساس قاعده زیر جایگزینی صورت می‌گیرد.

$$\begin{cases} (1,l) \rightarrow \begin{cases} (2,p),(0,q) \;\; if \;\; Up \gg Down \\ (0,q),(2,p) \;\; if \;\; Up \ll Down \end{cases} \\ (3,l) \rightarrow \begin{cases} (2,p),(4,q) \;\; if \;\; Up \gg Down \\ (4,q),(2,p) \;\; if \;\; Up \ll Down \end{cases} \\ (5,l) \rightarrow \begin{cases} (4,q),(6,p) \;\; if \;\; Up \gg Down \\ (6,p),(4,q) \;\; if \;\; Up \ll Down \end{cases} \\ (7,l) \rightarrow \begin{cases} (0,q),(6,p) \;\; if \;\; Up \gg Down \\ (6,p),(0,q) \;\; if \;\; Up \ll Down \end{cases} \end{cases} \qquad (۶-۴)$$

مقادیر p و q بر مبنای شیب خط L تعیین شده و مجموع آن‌ها بایستی برابر l شود. به طور مثال اگر مقدار l برابر ۱۲ و شیب خط L برابر ۰/۵ باشد داریم:

$$\begin{cases} {}^{p}/_{q} = |0.5| \\ p + q = 12 \end{cases} \Rightarrow \; p = 6 \, , q = \; 6$$

اگر تعداد نقاط دو مجموعه Up و $Down$ نزدیک به هم باشد، در این حالت از دو جهت اصلی قبل و بعد از جهت قطری در رشته F برای تعیین ترتیب قرار گیری جهت های افقی و عمودی جایگزین استفاده می‌کنیم.

مجموعه نهایی F در واقع متناظر با ویژگی های استخراجی شده از داده ورودی است که در مرحله ساخت پایگاه دانش و همچنین مرحله شناسایی از آن استفاده خواهیم کرد. ما از این به بعد مجموعه F را به شکل یک ماتریس با دو بردار ستونی به صورت $[d \quad l]$ نشان خواهیم داد که ستون اول را بردار ویژگی جهتی و ستون دوم را بردار ویژگی طول یا اندازه خواهیم نامید.

$$F = [d \quad l], d^T = [d_1 \; d_2 \; \dots \; d_c] \; , l^T = [l_1 \; l_2 \; \dots \; l_c] \qquad (۷-۴)$$

۴-۴- پایگاه دانش

در روش پیشنهادی هسته گام شناسایی بر مبنای تکنیک انطباق الگو بنا نهاده شده است. همانطور که در فصل سه در مبحث انطباق الگو مطرح شد، در این مبحث نیاز به الگوهای مرجع وجود دارد که این الگوهای مرجع در سیستم تشخیص دستنوشته برخط پیشنهادی اَشکال نوشتاری رایج حروف در مکان‌های مختلف زیرکلمه هستند. لذا در این بخش به نحوه استخراج الگوهای مرجع برای تولید پایگاه دانش می پردازیم.

اجازه دهید برای روشن شدن بیشتر موضوع نگاهی دوباره به مسئله شناسایی بیاندازیم. قبلاً گفته شد که در زبان فارسی ۳۲ حرف الفبا وجود دارد، اما بدنه تعداد زیادی از حروف الفبا مشابه یکدیگر بوده و تنها علامت متفاوتی دارند. بنابراین ابتدا به دسته بندی حروف بر اساس بدنه ی آنها می پردازیم که در جدول ۴-۱ این دسته بندی نشان داده شده است.

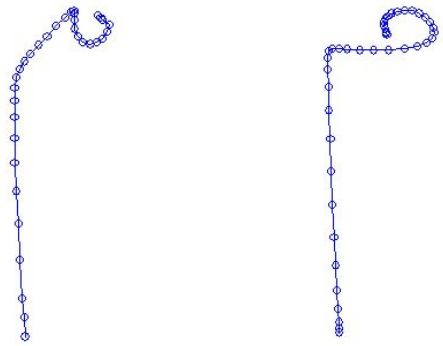
جدول ۴-۱: دسته بندی بدنه حروف فارسی

۱	ا آ	۱۰	ف
۲	ب پ ت ث	۱۱	ق
۳	ج چ ح خ	۱۲	ک گ
۴	د ذ	۱۳	ل
۵	ر ز ژ	۱۴	م
۶	س ش	۱۵	ن
۷	ص ض	۱۶	و
۸	ط ظ	۱۷	ه
۹	ع غ	۱۸	ی

از طرفی هر حرف بر اساس مکان قرارگیری در زیرکلمه که می تواند ابتدا، وسط، انتها و یا به شکل جداگانه باشد، شکل نوشتاری متفاوتی می‌تواند بگیرد. بنابر این در کل ۷۲ = ۱۸×۴ گروه خواهیم داشت، که مواردی از این گروه‌ها با هم مشابه هستند که اقدام به ادغام آن‌ها می‌کنیم. مثلاً حرف "ف" و "ق" در دو مکان انتها و جداگانه شکل نوشتاری متفاوتی دارند اما اگر در ابتدا یا وسط زیرکلمه قرار بگیرند بدنه آن‌ها شبیه یکدیگر است، به همین دلیل دو گروه ابتدا و وسط از این دو دسته را با هم ادغام می کنیم.

با این حال در گروه‌های تعریف شده باز هم شکل نوشتاری تعدادی از گروه‌ها مثل گروه‌های مکان انتهایی و جداگانه برای هر دسته شباهت زیادی به هم دارند، ولی به دلیل وجود خط پیوند[1] و تغییر ایجاد شده ناشی از آن ما به ترکیب این گروه‌ها نمی‌پردازیم.

در این جا بایستی به این نکته اشاره کرد که برای هر گروه الزماً یک الگوی مرجع کافی نیست و به دلیل تنوع در نوشتار حروف در مکان‌های مختلف زیرکلمه توسط نویسندگان مختلف و ازطرفی مستقل از نویسنده بودن سیستم پیشنهادی، برای هر گروه عموماً چندین الگو وجود خواهد داشت. برای روشن شدن منظور ما از تنوع در نوشتار حروف، دو نوع نوشتاری مختلف یک حرف نمونه در شکل ۴-۵ نشان داده شده است.



شکل ۴-۵: دو نمونه شکل نوشتاری گوناگون برای حرف "م"

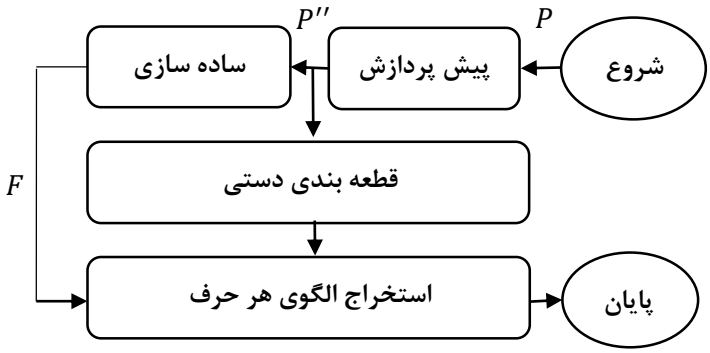
برای استخراج الگوهای مرجع هر گروه می توان دو رویکرد متفاوت داشت، در رویکرد اول می توان به جمع آوری دستی الگوهای مرجع پرداخت، این کار در صورتی که تعداد الگوهای مرجع زیاد نباشد عملی تر است، مثل وقتی که مسئله شناسایی به تنها شناسایی حروف جداگانه ساده شود و یا سیستم تشخیص دستنوشته تک کاربره باشد. در این حالت یک شخص ماهر با مشاهده نحوه نوشتاری حروف جداگانه و یا برای سیستم تک کاربره با بررسی دستخط کاربر، اقدام به استخراج الگوهای مرجع می کند.

مشکل این رویکرد آن است که برای مسئله اصلی ما یعنی یک سیستم تشخیص دستنوشته برخط مستقل از نویسنده که عنوان کردیم دارای ۷۲ گروه بوده و هر گروه می تواند چندین الگوی مرجع متفاوت داشته باشد، ارائه این تعداد الگو مرجع کاری طاقت فرسا خواهد بود، لذا می توان از رویکرد دوم استفاده کرد. در این رویکرد، برای

[1] Ligature

استخراج الگوهای مرجع از خود داده های ورودی استفاده می کنیم. در واقع به شکل خودکار از روی پایگاه داده، انواع مختلف رایج نوشتار حروف را به عنوان الگوهای مرجع استخراج می کنیم.

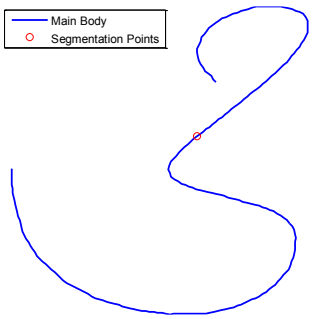
در روش پیشنهادی ما از این رویکرد استفاده نموده ایم که نحوه انجام آن در بلوک دیاگرام شکل ۴-۶ نشان داده شده است. در ادامه به توضیح مراحل این کار می پردازیم.



شکل ۴-۶: روند استخراج الگوهای مرجع بدنه حروف

به دلیل عدم وجود پایگاه داده ای به صورت آماده برای حروف ابتدا، وسط و انتها در اولین مرحله تولید پایگاه دانش به قطعه بندی دستی زیرکلمات پایگاه داده می پردازیم تا نمونه هایی از تمام حروف در مکان های مختلف داشته باشیم. قطعه بندی دستی بر روی بدنه زیرکلماتی انجام می شود که قبلا بر روی آنها پیش پردازش صورت گرفته است و اندیس[1] نقاط قطعه بندی، به عنوان اطلاعات حقیقی پایه[2] ذخیره می شوند. نمونه ای از قطعه بندی در شکل ۴-۷ نشان داده شده است.

[1] Index
[2] Ground Truth



شکل ۴-۷: نقاط قطعه بندی

در نهایت بعد از انجام گام بر روی زیرکلمه، حال هر زیر مجموعه متناظر با هر حرف از F به عنوان الگوی مرجع گروه آن حرف ذخیره می شود. به طور مثال برای زیرکلمه "حی" پیش پردازش شده در شکل ۴-۷ نقطه ی ۸۰ ام نقطه ی قطعه بندی است، مجموعه F استخراجی از کل زیرکلمه نیز پس از ساده سازی بشکل ماتریس در زیر آمده است که ستون اول ماتریس F جهات چهارگانه اصلی و ستون دوم، تعداد نقاط یا اندازه متناظر با جهت خود است.

$$\begin{bmatrix} 4 & 3 \\ 2 & 12 \\ 0 & 23 \\ 6 & 28 \\ 4 & 23 \\ 6 & 4 \\ 0 & 29 \\ 6 & 21 \\ 4 & 50 \\ 2 & 33 \end{bmatrix}$$

در این جا با توجه به نقطه ۸۰ ام به عنوان نقطه جدایی دو حرف در زیرکلمه، الگوی استخراج شده برای گروه حرف "ح" شروع برابر f_1 و برای گروه حرف "ی" انتهایی برابر f_2 خواهد بود.

$$F \xrightarrow{Split\ from\ point\ 80th} \begin{cases} f_1 = \begin{bmatrix} 4 & 3 \\ 2 & 12 \\ 0 & 23 \\ 6 & 28 \\ 4 & 14 \end{bmatrix} \\ f_2 = \begin{bmatrix} 4 & 9 \\ 6 & 4 \\ 0 & 29 \\ 6 & 21 \\ 4 & 50 \\ 2 & 33 \end{bmatrix} \end{cases}$$

این روند استخراج الگو برای تمام داده های پایگاه داده تکرار می شود. در مواردی که الگوی استخراج شده از لحاظ ستون جهات، یک الگوی تکراری بوده و از قبل در گروه مورد نظر موجود باشد، ستون اندازه الگو (ستون دوم) موجود در گروه با میانگین گیری به روز می شود که برای عملی شدن این کار تعداد تکرار هر الگوی هر گروه نیز ذخیره می شود.

بعد از اتمام فرایند تولید ، پایگاه دانش حاصل آماده استفاده است، هرچند عدم حذف کامل نویز، قطعه بندی دستی غیر ایده ال، نوشتار غلط نویسندگان و مواردی از این جمله همگی باعث می شوند تا الگوهایی وارد پایگاه دانش شوند که تاثیر نامناسبی بر عملکرد سیستم می گذارند، بهمین دلیل برای داشتن یک پایگاه دانش کوچک و کارا، بازبینی پایگاه دانش امری اجتناب ناپذیر است. در بازبینی انجام شده تمامی الگوهای نامتعارف و الگوهای مشکل ساز (بطور مثال الگوی [... 4] برای گروه ۲) حذف شدند. در موارد بسیاری در گروه ها الگوهایی وجود دارند که بردار ویژگی جهتی بسیار مشابهی دارند (که تنها در یک جهت ابتدایی یا انتهایی با هم تفاوت دارند) که به منظور کوچک کردن پایگاه دانش تنها یکی از الگو ها حفظ و بقیه را حذف می کنیم.

۴-۵- شناسایی

بعد از استخراج ماتریس ویژگی از الگوی ورودی ناشناخته، وارد گام شناسایی می شویم. خروجی این گام لیستی از چندین رشته پیشنهادی خواهد بود که هر رشته از بدنه حروف تشکیل شده است. روال کلی شناسایی بدین نحو است که تعداد مشخصی از سازگارترین الگوهای مرجع با ابتدای داده ورودی منطبق می شوند. سپس از ادامه

بهترین انطباق های صورت گرفته همین روند تکرار شده و این کار تا رسیدن به انتهای رشته داده ورودی تکرار می‌شود.

برای انجام این کار ما از یک الگوریتم بازگشتی[1] استفاده می کنیم که بصورت شبه کد در زیر آمده است.

```
function Recognition(SubWord,Start)
 if Start < F_len+Δ & Start > F_len−Δ
      Save SubWord;
 end
for i=Start: F_len
   Pos ← Determine Postion(Start,Middle,End,Isolate) of sub string
   Offers ←Calculate distances betwee F(Start:i,:) with all Models
   in position Pos;
 end
 Best_Matched ← Select best matched bodies with minimum
 distance from Offers.
 End_of_Match ← End of sub string related to best matched bodies.
 for i=1:size(Best_Matched)
    Recognition(Concate( SubWord , Best_Bodies(i)),End_of_Match(i)+1);
 end
end
```

ورودی الگوریتم شناسایی، ماتریس $F_{n\times 2}$ استخراج شده از الگوی ورودی ناشناخته است که از دو بردار $d_{n\times 1}$ و $l_{n\times 1}$ تشکیل شده است. ماتریس الگوهای مرجع نیز در ساختار Models ذخیره شده اند که خانه (i,j) در این ساختار حاوی مدل های مرجع بدنه حروف دسته i ام در مکان j است. $j = 1,2,3,4$ متناظر با چهار مکان قرارگیری حرف در زیرکلمه یعنی اول، وسط، انتها و یا حالت جداگانه است.

الگوریتم به انطباق زیررشته های $d_{Start:i}, i = Start,..,n$ (منظور از $d_{Start:i}$ زیررشته ای از رشته d با اندیس‌های از $Start$ تا i است و مقدار اولیه $Start$ برابر یک است) با تمام الگوهای مرجع در مکان مناسب می‌پردازد. این که زیر رشته با الگوهای کدام مکان مقایسه شود بر اساس اندیس های $Start$ و i تعیین می شود. بدین شکل که اگر $Start = 1$ و $i = n$ باشد، با الگوهای مرجع جداگانه مقایسه می شود. اگر تنها $Start = 1$ باشد، با الگوهای شروع، و اگر $i = n$ باشد با الگوهای انتها و در غیر اینصورت با الگوهای مرجع مکان وسط مقایسه می شود. سپس نقطه $Start$ برابر انتهای چند زیر رشته ای که بهترین انطباق را داشته اند قرار گرفته،

[1] Recursive

حرف متناظر با الگوی منطبق شده ذخیره شده و همین روند از نقطه های $Start$ جدید ادامه می یابد تا به انتهای رشته برسیم.

تعیین تعداد بهترین انطباق ها رابطه مستقیمی با زمان اجرا و درصد تشخیص نهایی خواهد داشت طوریکه هرچه تعداد بیشتری بهترین انطباق استفاده شود، درصد تشخیص بالاتر رفته ولی باعث بالارفتن زمان پردازش نیز خواهد شد.

برای اندازه گیری مشابهت هر زیررشته جهتی و زیررشته اندازه متناظر با آن از ماتریس ویژگی به صورت $[d_1 \quad l_1]$ با ماتریس ویژگی هر الگوی مرجع $[d_2 \quad l_2]$ از الگوریتم زیر استفاده می شود. در این الگوریتم از بردارهای l_1 و l_2 به عنوان توابع وزن یا جریمه عملیات های سه گانه درج، حذف و جایگزینی استفاده می شود.

```
Algorithm Minimum-Edit-Distance
 INPUT = d1,l1,d2,l2;
 OUT = MinDis;
 D = [d1Length, d2Length];
 D(0,0)=0;
for i=1 to d1Length
  D(i,0)=D(i-1,0)+l1(i);
end
for i=1 to d2Length
  D(0,i)=D(0,i-1)+l2(i);
end
for i=1 to d1Length
  for j=1 to d2Length
    if d1(i) == d2(j) then
     ReplacePenalty = abs(l1(i)-l2(j));
    else
     ReplacePenalty = l1(i)+l2(j)
   endif
    DeletPenalty = l1(i);
    InsertPenalty = l2(j);
    D(i,j) = min(D(i-1,j-1)+ReplacePenalty,
              D(i-1,j)+DeletePenalty,
              D(i,j-1)+InsertPenalty);
   endfor
 endfor
 MinDis = D(d1Length, d2Length);
end
```

ورودی این الگوریتم، رشته V' مربوط به داده ورودی به فرمت یک بردار جهتی d_1 و یک بردار اندازه متناظر l_1 می‌باشد که عنصر i ام از بردار l_1 طول جهت عنصر i ام از بردار d_1 را مشخص می‌کند. بطور مشابه یک الگو مرجع

بصورت بردار جهتی d_2 و اندازه l_2 می‌باشد. طول بردارهای d_1 و d_2 بترتیب $d_1Length$ و $d_2Length$ در نظر گرفته شده است. خروجی الگوریتم فاصله اصلاح مینیمم دو رشته جهتی d_1 و d_2 را نتیجه می‌دهد. در الگوریتم پیشنهاد شده در این مقاله بر خلاف روش استاندارد فاصله اصلاح مینیمم میزان جریمه برای اصلاح، حذف و اضافه کردن را برابر یک واحد در نظر نمی‌گیرد. بجای استفاده از جریمه واحد، بسته به طول جهت‌ها در الگوهای مورد بررسی جریمه جایگزینی ReplacePenalty، جریمه حذف DeletePenalty و جریمه اضافه کردن InsertPenalty از روی بردار های طول l_1 و l_2 محاسبه می شود. مطابق الگوریتم پیشنهادی اگر جهت‌های مورد مقایسه یکی باشند اختلاف طول آنها به عنوان جریمه در نظر گرفته می‌شود. اگر این جهت‌ها با هم یکی نباشند مجموع طول هر کدام به عنوان جریمه در نظر گرفته شده‌اند. در صورتی که بخواهیم یک عنصر رشته (یک جهت) را حذف یا اضافه کنیم به اندازه طول آن جهت جریمه در نظر گرفته می‌شود.

در این الگوریتم قبل از محاسبه فاصله اصلاح، بردار طول ویژگی نرمال می شود به شکلی که مجموع اندازه بردار طول برابر یک شود یعنی داریم:

$$l_i = \frac{l_i}{\sum_{i=1}^{m} l_i} \qquad (۸-۴)$$

اینکار باعث مستقل شدن روش نسبت به اندازه‌های نوشتاری متفاوت نویسندگان می شود و از طرفی باعث می‌شود که فاصله اصلاح محاسبه شده همیشه به مقداری در بازه صفر تا دو محدود شود. این مزیت باعث می‌شود تا برای انتخاب بدنه های منطبق شده با داده ورودی، بتوان از مقدار آستانه نیز استفاده کرد.

در نهایت لیست زیرکلمات پیشنهادی بر اساس مقدار میانگین فاصله اصلاح حروف تشخیص داده شده به ترتیب صعودی مرتب می شود.

۴-۶- پس پردازش

خروجی گام شناسایی لیستی پیشنهادی از رشته بدنه زیرکلمات است که می توان بسیاری از این پیشنهادات را بر اساس اطلاعات موجود دیگر در دستنوشته مثل علامت های زیرکلمه و همین طور اطلاعات مدل زبان حذف کرد. در زبان فارسی هرچند از لحاظ تئوری امکان ترکیب تمام حروف الفبا وجود دارد، اما در عمل بسیاری از ترکیب‌های حروف مورد استفاده نیستند. ما از این ویژگی که به آن مدل زبان گویند استفاده می‌کنیم. مدل زبان

می‌تواند یک فرهنگ لغت باشد که در این تحقیق همان‌طور که قبلاً نیز اشاره شد از خود زیرکلمات پایگاه داده بعنوان مدل زبان استفاده شده است. عملیات فیلتر کردن زیرکلمات پیشنهادی در دو بخش صورت می‌گیرد که در ادامه به آن‌ها می‌پردازیم.

۴-۶-۱- فیلتر بر اساس بدنه زیرکلمه

برای استفاده از مدل زبان، ابتدا برای یک بار زیر بدنه تمام زیرکلمات موجود در پایگاه داده در یک نوع ساختمان داده با دسترسی مستقیم ذخیره می شوند. سپس تمام زیرکلمات پیشنهادی گام شناسایی یک به یک بررسی شده و درصورتی که بدنه آن ها در ساختمان داده وجود نداشته باشد حذف می شوند و درغیر اینصورت زیرکلمات متناظر با رشته زیر بدنه پیشنهادی به لیستی اضافه می شوند.

بطور مثال برای زیرکلمه ورودی "حی" که قبلا شکل آن نشان داده شد، تعدادی از پیشنهادات گام شناسایی عبارتند از:

"محا"،"حما"،"فحا"،"حی"،"ححا"،"طی"،" طحا"،" می"،"فی"،"حن"،"طن"...

بایستی به این نکته توجه داشت که در پیشنهادات بالا، هر حرف نماینده یک گروه است. یعنی مثلاً "ححا" می‌تواند نماینده هر کدام از زیرکلمات "خجا"و "حجا"و ... باشد.

نتیجه اعمال این فیلتر آن است که علاوه بر حذف بدنه زیرکلمات ناموجود در فرهنگ لغت، لیست پیشنهادی رشته بدنه حروف، به لیستی از زیرکلمات تبدیل می شود.

در این جا شاید این ابهام برای خوانند ایجاد شود که در این حالت شناسایی سیستم تنها به زیرکلمات موجود در فرهنگ لغت محدود می شود و بنابراین با سیستم های با رویکرد کلی نگر، که تنها قادر به شناسایی داده های موجود در پایگاه داده خود بودند، تفاوت چندانی ندارد.

برای روشن شدن این ابهام بایستی به این نکته توجه داشت که هر چند محدود شدن دایره زیرکلمات قابل شناسایی به فرهنگ لغت گفته درستی است، اما باید بدانیم که نقش فرهنگ لغت در سیستم پیشنهادی تنها به عنوان یک فیلتر است و برای شناسایی زیرکلمات ناموجود در فرهنگ لغت کافی است تا آن ها براحتی به فرهنگ لغت اضافه شوند. این کار هیچ تاثیری منفی بر روی گام بازشناسی ندارد، زیرا به معنای بیشتر شدن تعداد کلاس های مسئله نیست.

۴-۶-۲- فیلتر بر اساس تعداد و مکان علامت ها

علامتهای دستنوشته همیشه حاوی اطلاعات با ارزشی در مورد دستنوشته هستند، اما دلایل بسیاری مثل شکل های نوشتاری مختلف نقاط و عدم دقت نویسنده در قرار دادن علامت ها در مکان صحیح آن ها، باعث می شود تا استفاده از علائم کاری مشکل باشد. در این تحقیق تلاشی برای شناسایی علامت ها صورت نگرفته اما ما از تعداد و همچنین مکان قرارگیری آن ها نسبت به بدنه زیرکلمه که می تواند در بالا یا پایین آن باشد، به عنوان فیلتری مفید برای کاهش زیرکلمات پیشنهادی گام شناسایی استفاده می کنیم.

تعیین محل نسبی علامت ها نسبت به بدنه زیرکلمه بر مبنای الگوریتم ارائه شده در [۳۰] صورت می گیرد. مجموعه نقاط مربوط به بدنه زیرکلمه P و مجموعه نقاط مربوط به یک حرکت علامت دو ورودی الگوریتم بوده و خروجی الگوریتم موقعیت علامت نسبت به بدنه خواهد بود که می تواند بالا یا پایین باشد.

نحوه کارکرد الگوریتم به این شکل است که اگر اولین نقطه حرکت علامت (به ترتیب نوشتن) پایین تر از اولین نقطه از بدنه که با علامت مذکور همپوشانی افقی دارد باشد، حرکت را زیر بدنه اصلی در نظر می گیریم، در غیر اینصورت، حرکت بالای بدنه اصلی است. درصورتیکه حرکت با بدنه حرف همپوشانی افقی نداشته باشد، نزدیکترین نقطه از نظر فاصله افقی با حرکت را در نظر می گیریم. برای رفع ابهام محل نقطه حروف "ج/چ" که نقطه با بدنه اصلی می تواند در دو محل همپوشانی افقی داشته باشد، کافی است که اولین محل همپوشانی (به ترتیب نوشته شدن) را در نظر بگیریم. همچنین اگر نشانه این حروف طوری گذاشته شود که فقط با قسمت پایین حروف همپوشانی داشته باشد، برای رفع این مشکل ۲۰ درصد آخر حرف درنظر گرفته نمی شود تا کشیدگی انتهای حرف به عنوان محل همپوشانی افقی با نقطه در نظر گرفته نشود. این الگوریتم برای تمام حرکت های داده ورودی ناشناخته (غیر از بدنه) انجام می شود و تعداد علامتهای بالا، UpStroke و پایین بدنه DownStroke زیرکلمه محاسبه می شود.

در روش پیشنهادی برای هر حرف الفبا در هر چهار موقعیت ممکن یک بیشینه و کمینه تعداد علامت برای هر دو مکان بالا و پایین حرف تعیین می‌شود این مقادیر در یک ماتریس ۲×۲ که آن را ماتریس محدوده می نامییم بشکل زیر ذخیره می شوند.

$$\begin{bmatrix} \text{بیشینه بالا} & \text{کمینه بالا} \\ \text{بیشینه پایین} & \text{کمینه پایین} \end{bmatrix}$$

بطور مثال حرف "ی" در مکان ابتدای زیرکلمه دو نقطه دارد، این دو نقطه توسط نویسندگان مختلف عموما با یک یا دو حرکت (دو نقطه جداگانه) نوشته می شود، در نتیجه محدوده تعداد علامت‌ها برای حرف "ی" برای چهار مکان مختلف حرف بشکل زیر خواهد بود.

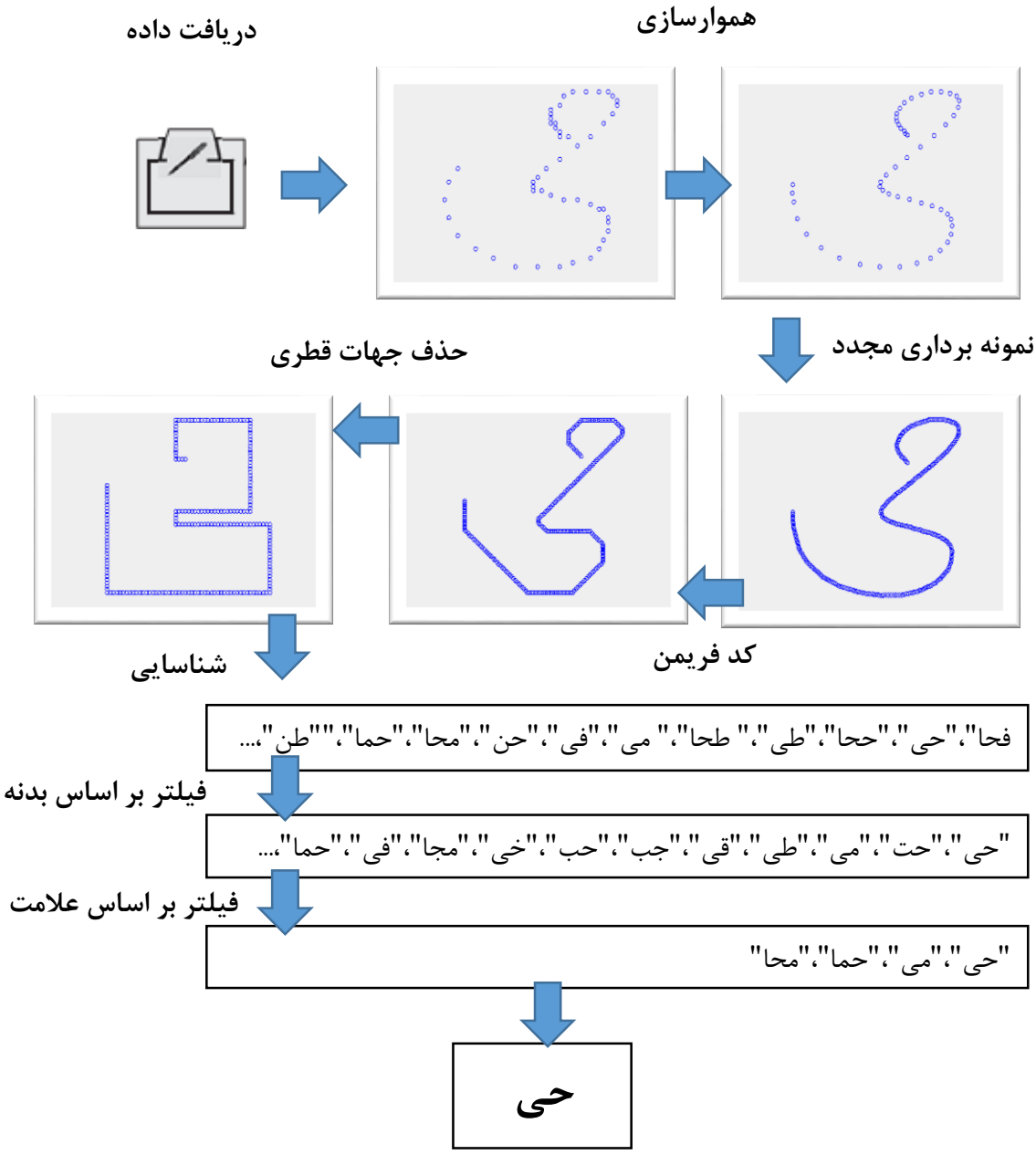
$$\text{ابتدا} \rightarrow \begin{bmatrix}0 & 0\\ 1 & 2\end{bmatrix},\ \text{وسط} \rightarrow \begin{bmatrix}0 & 0\\ 1 & 2\end{bmatrix},\ \text{انتها} \rightarrow \begin{bmatrix}0 & 0\\ 0 & 0\end{bmatrix},\ \text{جداگانه} \rightarrow \begin{bmatrix}0 & 0\\ 0 & 0\end{bmatrix}$$

برای هر زیرکلمه نیز بیشینه و کمینه علامت ها در بالا و پایین هر زیرکلمه بسادگی با جمع بیشینه و کمینه تعریف شده برای حروف سازنده زیرکلمه محاسبه می شود. بطور مثال برای زیرکلمه "تحقیقا" ماتریس تعداد علامت‌های زیرکلمه برابر خواهد بود با:

$$\text{"تحقیقا"} \rightarrow \begin{cases} \text{"ت" ابتدا} \rightarrow \begin{bmatrix}1 & 2\\ 0 & 0\end{bmatrix} \\ \text{"ح" وسط} \rightarrow \begin{bmatrix}0 & 0\\ 0 & 0\end{bmatrix} \\ \text{"ق" وسط} \rightarrow \begin{bmatrix}1 & 2\\ 0 & 0\end{bmatrix} \\ \text{"ی" وسط} \rightarrow \begin{bmatrix}0 & 0\\ 1 & 2\end{bmatrix} \\ \text{"ق" وسط} \rightarrow \begin{bmatrix}1 & 2\\ 0 & 0\end{bmatrix} \\ \text{"ا" انتها} \rightarrow \begin{bmatrix}0 & 0\\ 0 & 0\end{bmatrix} \end{cases} \Rightarrow \begin{bmatrix}1 & 2\\ 0 & 0\end{bmatrix} + \begin{bmatrix}0 & 0\\ 0 & 0\end{bmatrix} + \begin{bmatrix}1 & 2\\ 0 & 0\end{bmatrix} + \begin{bmatrix}1 & 2\\ 1 & 2\end{bmatrix} + \begin{bmatrix}1 & 2\\ 0 & 0\end{bmatrix} + \begin{bmatrix}0 & 0\\ 0 & 0\end{bmatrix} = \begin{bmatrix}4 & 6\\ 1 & 2\end{bmatrix}$$

در نهایت زیرکلمات پیشنهادی که UpStroke و DownStroke داده ورودی ناشناخته در بازه ماتریس محدوده آن ها نباشد حذف می شوند.

در شکل ۴-۸ با یک مثال فرایند تشخیص زیرکلمه ورودی ناشناخته "حی" مرور شده است. برای این ورودی تنها یک حرکت قلم وجود دارد، بنابر این علامتی برای این ورودی وجود ندارد. در این مثال به خوبی می توان پیچیدگی کار تشخیص دستنوشته را مشاهده کرد زیرا زیرکلمه ورودی شباهت زیادی به زیرکلمات زیادی مانند "حی"، "می" "محا"، "فحا"، "ححا" و ... را دارد و درنتیجه تشخیص ورودی در اولین پیشنهاد سیستم کار مشکلی است.



شکل۴-۸: یک مثال از فرایند تشخیص الگوی ورودی ناشناخته

فصل ۵

نتایج شبیه سازی و تحلیل آن‌ها

فصل ۵- نتایج شبیه سازی و تحلیل آن ها

۵-۱- شبیه سازی

در این فصل نتایج پیاده‌سازی روش پیشنهادی که در فصل قبل شرح داده شد، با نرم افزار MATLAB بر روی سخت افزار با مشخصات 4GB RAM DDR 3 و پردازنده Core i7 1.73 GHz ذکر شده است. تمامی این شبیه‌سازی‌ها بر روی پایگاه داده حروف و زیرکلمات دانشگاه تربیت مدرس [۸] که در [۳۱] بررسی شده است، انجام شده است. این پایگاه داده شامل بیش از ۱۰۰۰ زیر کلمه متداول در زبان فارسی، ارقام صفر تا نه و بیش از ۴۰۰۰ حرف جداگانه فارسی نوشته شده توسط ۱۲۸ نفر است. حروف جداگانه این پایگاه داده ۳۲ حرف فارسی (بهمراه نشانه‌ها)، حرف آ (با سرکش) و همین‌طور همزه بوده و از هر حرف حدوداً ۱۲۰ نمونه موجود است. مجموعه اطلاعات این پایگاه داده بصورت مجموعه نقاط مسیر حرکت قلم که هر نقطه با مختصات x و y آن مشخص می‌گردد، ذخیره شده‌اند.

نتایج شبیه سازی در سه بخش ارائه خواهد گردید و سپس با کارهای مشابه مقایسه شده است.. در بخش ۵-۲ نتایج روش پیشنهادی برای شناسایی حروف قطعه بندی شده ارائه خواهد شد. در بخش ۵-۳ نتایج شناسایی تنها حروف جداگانه فارسی خواهد آمد و در بخش ۵-۴ نتایج روش برای شناسایی زیرکلمه ذکر خواهد گردید.

۵-۲- نتایج شناسایی بدنه حروف قطعه‌بندی شده

در این بخش به بررسی عملکرد روش پیشنهادی برای شناسایی بدنه حروف قطعه‌بندی شده می‌پردازیم. برای این کار از پایگاه داده زیرکلمات تربیت مدرس که به شکل دستی قطعه‌بندی شده‌اند، استفاده شده است. پس از انجام مراحل پیش‌پردازش و ساده‌سازی بر روی هر حرف قطعه‌بندی شده در هر چهار مکان مختلف از زیرکلمه یعنی ابتدا، وسط، انتها و جدا، با استفاده از الگوریتم محاسبه فاصله اصلاح ارائه شده در فصل ۴ فاصله بدنه حرف ورودی با تمام گروه‌های الگوهای مرجع در آن موقعیت مقایسه می‌شود. در این جا بدلیل این که تنها بدنه حروف تشخیص داده می‌شود از فیلتر مکان و تعداد علامت‌ها استفاده نمی‌شود. در این حالت درصد شناسایی بدنه هر حرف در nTop پیشنهاد اول که کمترین فاصله اصلاح را دارند، در جدول ۵-۱ نشان داده شده است.

جدول ۵-۱: نتایج شناسایی بدنه حروف قطعه بندی شده در مکان های مختلف

موقعیت	تعداد نمونه	nTop (%)		
		1	2	5
شروع	10494	79.57	87.72	94.17
وسط	10842	72.55	86.27	95.08
آخر	10494	75.19	87.71	96.56
جدا	396	79.54	93.93	97.72

۵-۳- نتایج شناسایی روش پیشنهادی بر روی حروف جداگانه

در این بخش نتایج بررسی عملکرد سیستم پیشنهادی بر روی ۳۲ حرف جدا فارسی (بهمراه حرف آ با سرکش) این پایگاه داده ذکر شده است. برای شناسایی حروف جداگانه نیازی به استفاده از الگوریتم بازگشتی ارائه شده در فصل ۴ نیست و تنها کافی است که از پیش پردازش، ساده سازی و الگوریتم محاسبه فاصله اصلاح ارائه شده در فصل ۴ برای مقایسه نمونه ورودی با الگوهای مرجع حروف جدا، و همین طور فیلتر مکان و تعداد علامت ها استفاده شود. برای استخراج الگوهای مرجع حروف جدا نیز از داده های آموزشی همین پایگاه داده استفاده شده است.

برای این کار، دو گروه بندی بر روی حروف جدا فارسی انجام می شود. در گروه بندی اول حروف بر اساس محل قرارگیری علامت‌هایشان به سه گروه بالا، پایین و بدون نشانه تقسیم می‌شوند. دسته‌بندی حروف در این سه گروه در جدول ۵-۲ نشان داده شده است.

جدول ۵-۲: تقسیم بندی الفبا بر اساس مکان علامت حروف

گروه علامت بالا	آ،ت،ث،خ،ذ،ز،ژ،ش،ض،ط،ظ،غ،ف،ق،ک،گ،ن
گروه بدون علامت	ا،ح،د،ر،س،ص،ط،ع،ک،ل،م،و،ه،ی
گروه علامت پایین	ب،پ،ج،چ

در موارد نادری بدلیل شکل نوشتاری متفاوت نویسندگان یک حرف می‌تواند در بیش از یک گروه قرار گیرد. بطور مثال حروفی مثل "ک" و یا "ط" توسط بعضی نویسندگان بدون نشانه و بشکل پیوسته در یک حرکت قلم نوشته می‌شوند، لذا این دو حرف می‌توانند در گروه حروف با نشانه بالا و همین‌طور در گروه حروف بدون نشانه قرار گیرند.

در گروه بندی دوم حروف بر اساس شباهت شکل بدنه حرف دسته بندی می شوند. این گروه بندی نیز در جدول ۵-۳ نشان داده شده است.

جدول ۵-۳: گروه بندی بر اساس بدنه حروف

ا،آ	۱
ب،پ، ت،ث،ک،گ	۲
ج،چ،ح،خ	۳
د،ذ	۴
ر،ز،ژ	۵
س،ش	۶
ص،ض	۷
ط،ظ	۸
ع،غ	۹

ف	۱۰
ق	۱۱
ل	۱۲
م	۱۳
ن	۱۴
و	۱۵
ه	۱۶
ی	۱۷

درنهایت باتوجه به گروه‌بندی های ذکر شده، تعداد کلاس‌های قابل شناسایی روش پیشنهادی برابر ۲۶ کلاس خواهد بود که در جدول ۵-۴ آمده اند.

جدول ۵-۴: کلاس های قابل شناسایی

ا	۱
آ	۲
ب،پ	۳
ت،ث،ک،گ	۴
ج،چ	۵
ح	۶
خ	۷
د	۸
ذ	۹
ر	۱۰
ز،ژ	۱۱
س	۱۲
ش	۱۳

ص	۱۴
ض	۱۵
ط،ظ	۱۶
ع	۱۷
غ	۱۸
ف	۱۹
ق	۲۰
ل	۲۱
م	۲۲
ن	۲۳
و	۲۴
ه	۲۵
ی	۲۶

چنانچه هر گروه بدنه تنها توسط یک الگوی متداول آن در پایگاه دانش مدل گردد درصد تشخیص برای هر کلاس جدول ۵-۴ در جدول ۵-۵ نشان داده شده است.

جدول ۵-۶: نتایج شبیه سازی با استفاده از چند الگو

گروه	درصد	گروه	درصد
1	100	14	95
2	99	15	99
3	100	16	96
4	94	17	97
5	99	18	99
6	95	19	81
7	99	20	90
8	90	21	94
9	82	22	95
10	96	23	95
11	94	24	98
12	96	25	90
13	94	26	94

95.2% :درصد شناسایی کل

جدول ۵-۵: نتایج شبیه سازی با استفاده از تنها یک الگو

گروه	درصد	گروه	درصد
1	98	14	97
2	99	15	99
3	100	16	83
4	85	17	86
5	99	18	87
6	90	19	75
7	94	20	89
8	91	21	89
9	88	22	91
10	89	23	95
11	90	24	94
12	85	25	83
13	75	26	89

90.2% :درصد شناسایی کل

نتایج بدست آمده بخوبی بیانگر این است که روش پیشنهادی قابلیت خوبی در شناسایی حروف دارد. هرچند واضح است که با توجه به شکل‌های متفاوت نوشتاری که نمونه ای از آن در شکل ۴-۵ نشان داده شد، بایستی از الگوهای بیشتری برای مدل‌کردن بدنه بعضی از گروه‌ها استفاده نمود. در این آزمایش اگر از تعداد کافی مدل برای مدل‌کردن بدنه گروه ها استفاده شود، نتایج شناسایی روش پیشنهادی مطابق جدول ۵-۶ می‌باشد.

مقایسه جدول ۵-۵ و ۵-۶ نشان می‌دهد چنانچه از چند الگو برای مدل‌کردن بدنه یک گروه استفاده شود نتایج شناسایی بهبود قابل توجهی دارد.

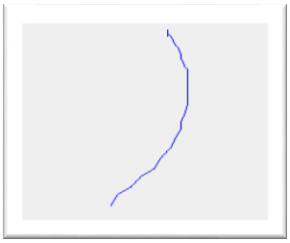
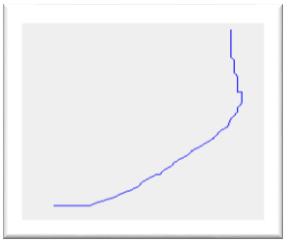
برای بررسی جزییات بیشتر ماتریس تداخل[1] در شناسایی حروف را در جدول ۵-۷ نشان داده‌ایم. عنصر سطر i و ستون jام در این ماتریس برابر با تعداد نمونه‌هایی از کلاس i است که اشتباهاً درکلاس j طبقه‌بندی شده است. بنابراین عناصر خارج قطر اصلی در این ماتریس اشتباهات رخ داده در طبقه‌بندی را نشان می‌دهد. می‌توان دید که این اشتباهات عموماً مربوط به نمونه‌هایی از گروه‌هایی است که بدنه آن‌ها پس از ساده‌سازی شباهت زیادی به یکدیگر دارند. در بسیاری موارد حتی یک شخص بیننده نیز در تشخیص این حروف مرتکب خطا می‌شود.

[1] Confusion Matrix

جدول ۵-۷: ماتریس تداخل

Group	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24	25	26
1	120	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	4	0	0	0	0
2	0	118	0	0	0	0	0	0	1	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
3	0	0	240	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
4	0	0	0	445	0	0	0	0	2	2	10	0	0	0	0	0	0	0	13	4	0	0	0	0	2	0
5	0	0	0	0	234	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
6	0	0	0	0	0	114	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
7	0	0	0	0	2	0	117	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
8	0	0	0	0	0	0	0	110	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	2	0	0
9	0	0	0	0	0	0	0	0	98	0	2	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
10	0	0	0	0	0	0	0	8	0	118	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1
11	0	0	0	7	0	0	0	0	16	0	227	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0
12	0	0	0	0	0	0	0	0	0	0	0	116	0	4	0	0	0	0	0	0	0	0	0	0	0	1
13	0	0	0	1	0	0	0	0	0	0	0	0	112	0	0	0	0	0	0	0	0	0	0	0	0	0
14	0	0	0	0	0	0	0	0	0	0	0	0	0	116	0	0	0	0	0	0	0	0	0	0	0	0
15	0	0	0	1	0	0	0	0	1	0	0	0	0	0	118	0	0	0	0	0	0	0	0	0	0	0
16	0	1	0	2	0	1	0	0	1	0	0	0	0	0	0	232	0	0	0	0	0	0	0	0	1	0
17	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	117	0	0	0	0	0	0	0	0	0
18	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	115	1	3	0	0	0	0	0	0
19	0	0	0	5	0	0	0	0	0	0	1	0	0	0	0	1	0	0	99	0	0	0	0	0	0	0
20	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	6	109	0	0	2	0	0	0
21	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	114	1	0	0	4	1
22	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	115	0	0	0	0
23	0	0	0	8	0	0	0	0	0	0	0	0	6	0	0	0	0	0	2	5	0	0	114	0	0	0
24	0	0	0	0	0	0	0	2	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	120	1	0
25	0	0	0	0	0	0	0	1	0	1	0	2	0	0	0	1	0	0	0	0	4	0	0	0	107	1
26	0	0	0	3	0	0	0	0	0	0	0	2	0	1	0	0	1	0	0	0	0	0	0	0	2	115

چند نمونه از اشتباهات طبقه‌بندی نیز در سیستم پیشنهادی در شکل ۵-۱ نشان داده شده است.


شکل ۵-۱: شکل سمت راست. حرف "د" که اشتباهاً "ر" تشخیص داده شده است. شکل وسط. بدنه حرف "گ" که اشتباهاً "ز" تشخیص داده شده است. شکل سمت چپ. حرف "ک" که اشتباهاً "ی" تشخیص داده شده است.

۵-۴- نتایج شناسایی روش پیشنهادی بر روی زیرکلمات

نتایج تشخیص زیرکلمات پایگاه داده [۸] برای زیرکلمات با تعداد حرف مختلف بصورت جداگانه در جدول ۵-۸ نشان داده شده است.

جدول ۵-۸: نتایج تشخیص زیرکلمات

مجموعه مورد شناسایی	تعداد نمونه	گزینه اول(٪)	۳ گزینه اول(٪)	میانگین زمانی (ثانیه)
زیرکلمات یک حرفی	۳۸۵	۵۶/۳۶	۹۱/۴۳	۰/۱۵
زیرکلمات دو حرفی	۲۸۹۳	۶/۸۳	۷۴/۷۳	۰/۴۳
زیرکلمات سه حرفی	۵۰۱۸	۸/۸۳	۶۴/۱۹	۰/۵۹
زیرکلمات چهار حرفی	۱۹۶۳	۱۰/۷۵	۴۸/۵۵	۳/۴۲
زیرکلمات پنج حرفی	۵۱۱	۱۳/۱۱	۳۵/۷۹	۱۲/۸۳
زیرکلمات شش حرفی	۱۰۹	۳/۶۷	۲۱/۱۰	۲۴/۶۵
زیرکلمات هفت حرفی	۱۵	۶/۶۷	۶/۶۷	۳۳/۶۷

نتیجه کلی بدست آمده برای سه گزینه اول برابر است با ۶۳/۲۹٪. نتایج جدول ۵-۸ نشان می دهد که با افزایش تعداد حروف درصد شناسایی کاهش چشمگیری پیدا کرده و زمان پردازش افزایش می یابد.

۵-۵- مقایسه نتایج

در این قسمت به مقایسه نتایج بدست آمده روش پیشنهادی با کارهای مشابه بیان شده در فصل دو می پردازیم. هرچند مقایسه کارها با پایگاه داده های متفاوت چندان منطقی نیست، اما نتایج صرفاً جهت اطلاع آورده شده اند.

۵-۵-۱ مقایسه نتایج شناسایی حروف جدا

در جدول ۵-۹ نتایج کارهای مشابه در زمینه تشخیص حروف جدا با روش پیشنهادی مقایسه شده است.

جدول ۵-۹: مقایسه تشخیص حروف جدا

مرجع	نتیجه (٪)	روش	پایگاه داده
[۲۱]	۹۴/۹	HMM	حروف برخط دانشگاه تربیت مدرس
[۱۹]	۹۰/۲۷	FLVQ	حروف برخط دانشگاه تربیت مدرس
[۲۰]	۹۳/۳	NN	حروف برخط دانشگاه تربیت مدرس
[۱۸]	۹۴	درخت تصمیم گیری	حروف برخط دانشگاه تربیت مدرس
[۳۰]	۹۰/۱۷ برای تشخیص گروه حرف	HMM	حروف برخط دانشگاه تربیت مدرس
روش پیشنهادی	۹۵/۲ برای تشخیص گروه حرف	ساده سازی به همراه انطباق الگو	حروف برخط دانشگاه تربیت مدرس

نتایج جدول ۵-۹ بخوبی کارایی گام ساده سازی و الگوریتم پیشنهادی محاسبه فاصله اصلاح را نشان می دهد. روش پیشنهادی از چند جنبه با روش‌های موجود متفاوت است که مهم‌ترین آن، مدل محور بودن سیستم است که نیاز آن به آموزش و داده‌های آموزشی را حذف می‌کند. از طرفی ساده سازی انجام شده در بازنمایی الگوها باعث می‌شود تا بی‌ثباتی‌های مربوط به دستنوشته به نحو موثری از نوشتار حذف شود.

۵-۵-۲- مقایسه شناسایی زیرکلمات

در این قسمت روش پیشنهادی بمنظور شناسایی زیرکلمات با کارهای مشابه با رویکرد تجزیه ای مقایسه شده است. این نتایج در جدول ۵-۱۰ نشان داده شده است.

جدول ۵-۱۰: مقایسه روشهای شناسایی زیرکلمات

مرجع	نتیجه	روش	پایگاه داده
[۲۳]	٪۹۸/۴۹ برای فرهنگ لغت ۵۰۰۰ کلمه ای	HMM	۶۲۲۰ زیرکلمه از ۸۰۰ کلمه عربی نوشته شده توسط ۱۰ نفر
[۲۷]	٪۷۸ بدون فرهنگ لغت و ٪۹۶ با فرهنگ لغت	FEM	۱۲۵۰ کلمه از کتاب‌های اول دبستان نوشته شده توسط ۱۰ بزرگ‌سال و ۱۰ دانش آموز
[۳۰]	٪۳۷/۸۵ برای گزینه اول، ٪۷۳/۳۴ برای ۱۰ گزینه اول	HMM	پایگاه داده زیرکلمات و حروف برخط دانشگاه تربیت مدرس
روش پیشنهادی	٪ ۶۳/۲۹ برای ۳ گزینه اول	ساده سازی و انطباق الگو	زیرکلمات برخط دانشگاه تربیت مدرس

فصل ۶

نتیجه گیری و کارهای آینده

فصل ۶- نتیجه گیری و کارهای آینده

۶-۱- نتیجه گیری

در عصر حاضر استفاده از وسایل الکترونیکی با صفحات لمسی مثل تلفن‌های همراه هوشمند و تبلت‌ها به شدت در حال گسترش است. علت این امر نیاز بشر به اطلاعات و همچنین رابط کاربری کاربرپسندی[1] است که این وسایل نسبت به رابط‌های قدیمی دیگر مثل صفحه کلید ارائه می‌کنند. گسترش این وسایل، بازار وسیعی از برنامه‌های کاربردی مختص این وسایل را نیز ایجاد کرده است. در این میان دستخط به عنوان یک شیوه ارتباط مورد علاقه بشر برای ایجاد ارتباط با دیگران باعث شده است تا توجه بسیاری از محققان و شرکت‌های تجاری به تحقیق و ایجاد برنامه‌های کاربردی برای تشخیص دستنوشته برای این وسایل اقدام کنند. در این وسایل مختصات رقومی شده نقاط لمس شده در هر لحظه موجودند، به همین دلیل به این داده ها برخط گویند. در زمینه تشخیص دستنوشته برخط لاتین کارهای زیادی صورت گرفته است اما برای زبان فارسی و دیگر زبان‌ها با الفبای مشابه با زبان فارسی این کار به تازه‌گی آغاز شده است زیرا تشخیص دستنوشته فارسی به دلیل طبیعت پیوسته متن، وجود علامت‌های حروف و تغییر شکل حروف در موقعیت‌های مختلف پیچیدگی‌های بیشتری نسبت به دستنوشته لاتین دارد.

در تشخیص دستنوشته برخط دو رویکرد شناسایی مختلف وجود دارد، در رویکرد کلی‌نگر، هر زیرکلمه به عنوان یک الگوی تجزیه‌ناپذیر در نظر گرفته و اقدام به شناسایی آن می‌شود. ولی در رویکرد دوم، رویکرد جزئی‌نگر هر زیرکلمه ابتدا به اجزای بامعنی مثل حروف سازنده آن تجزیه می‌شود.

در این تحقیق یک سیستم تشخیص دستنوشته برخط با رویکرد جزئی‌نگر برای زبان فارسی ارائه شده است. در این سیستم ابتدا به پیش‌پردازش داده ورودی می‌پردازیم که عبارتند از: هموارسازی و نمونه‌برداری مجدد سپس برای حذف تغییرات در دستنوشته و استخراج ویژگی‌های مناسب برای بازنمایی زیرکلمه یک گام ساده سازی داده ورودی در سه مرحله تبدیل به کد فریمن، فشرده‌سازی و حذف جهات قطری انجام می‌شود. سپس شکل ساده شده داده ورودی به مرحله شناسایی منتقل می‌شود.

از طرفی برای تولید الگوهای مرجع نیز به دلیل این که پایگاه داده‌ای برای حروف در مکان‌های مختلف زیرکلمه وجود ندارد، ابتدا تعدادی از زیرکلمات به عنوان داده‌های آموزشی بصورت دستی قطعه‌بندی شده و سپس با انجام

[1] User friendly

دو گام پیش پرازش و ساده سازی، ویژگی های استخراجی از بدنه داده‌های آموزشی به عنوان الگوهای مرجع بدنه حروف در مکان‌های مختلف ذخیره می‌شوند.

در گام شبیه‌سازی با استفاده از یک الگوریتم انطباق الگو ما به جستجوی الگوهای مرجع در ماتریس ویژگی‌های داده ورودی می‌پردازیم تا حروف سازنده زیرکلمه پیدا شوند. خروجی گام شناسایی لیستی از رشته‌های تشکیل شده از بدنه حروف خواهد بود. در نهایت از دو فیلتر برای حذف پیشنهاداتی که در فرهنگ لغت وجود ندارند و پیشنهاداتی که مکان و تعداد علامت آن‌ها با داده ورودی همخوانی ندارد استفاده شده است.

شبیه سازی انجام شده بر روی پایگاه داده زیرکلمات برخط دانشگاه تربیت مدرس انجام شده است. در تهیه این پایگاه داده، تنها محدودیت نویسندگان نوشتن بدنه اصلی زیرکلمه قبل از گذاشتن حرکت‌های علامت‌ها بوده است. این پایگاه از دستخط بیش از صد نویسنده تهیه شده و به همین دلیل از تنوع بسیار بالایی برخوردار است، هرچند در تعدادی از نمونه های پایگاه داده، مشکلاتی وجود دارد که کار شناسایی را سخت می کند از جمله این مشکلات می توان وجود قلاب در ابتدا یا انتهای تعداد قابل توجهی از نمونه ها و نویز بالای موجود در داده ها را نام برد.

۶-۲- کارهای آینده

با بررسی نتایج ذکر شده در فصل قبل و تجربه بدست آمده در زمینه تشخیص دستنوشته برخط، برای بهبود روش پیشنهادی می‌توان مواردی را به عنوان کارهای آینده مطرح نمود که در زیر چند مورد ذکر شده است.

۱- **ارائه روشی برای حذف قلاب:** در پایگاه داده مورد استفاده تعداد قابل توجهی از نمونه‌ها دارای قلاب در ابتدا و یا انتهای خود هستند. وجود قلاب کار شناسایی را با مشکل روبرو می‌سازد. به همین دلیل درصورت استفاده از این پایگاه داده استفاده از یک الگوریتم کارا برای حذف قلاب مفید خواهد بود. عموما روش‌هایی که برای تشخیص قلاب برای زبان‌های لاتین پیشنهاد شده‌اند، این کار را بر مبنای دو اصل انجام می‌دهند. اول آن که اندازه قلاب کوچک بوده و درصد ناچیزی از زیرکلمه را تشکیل می‌دهد و دوما در نقطه انتهای قلاب یک زوایه شکست کوچک وجود دارد. هر چند بر اساس این دو شرط و شروط اصلاحی دیگری تلاشی برای حذف قلاب صورت گرفت اما این دو اصل متاسفانه نمی‌توانند مشکل قلاب را در این پایگاه داده را به شکل کارایی حل کنند. اول به دلیل این که اندازه قلاب‌ها در مواردی بسیار بزرگ است و در مقابل حروف

ابتدایی توسط بعضی نویسندگان بسیار کوچک نوشته می‌شوند و دوم آنکه شکل نگارش بعضی از حروف مثل "م" و "ح" در ابتدای زیرکلمه ذاتاً حالت قلاب مانند می‌تواند داشته باشد.

۲- **تهیه پایگاه داده** : هرچند پایگاه داده مورد استفاده در این تحقیق یکی از متنوع ترین پایگاه داده‌های موجود برای زبان فارسی است، اما بدلایلی ایجاد یک پایگاه داده جدید حس می شود، از جمله:

- وجود قلاب که قبلا در مورد آن صحبت کردیم.
- نویز بسیار زیاد در تعدادی از نمونه ها که حتی با هموارسازی نیز به خوبی قابل حذف نیستند.

این موارد به نظر همگی از مناسب نبودن سخت افزار بکارگرفته و عدم آشنایی نویسندگان به چگونگی استفاده از این سخت افزار ناشی شده است.

۳- **شناسایی علامت‌ها**: علامت‌ها در شناسایی کلمه برای افراد نقش بسزایی دارند. به دلیل این که شکل نوشتاری علامت‌ها تغییرات بسیار زیادی دارند در روش پیشنهادی تلاشی برای شناسایی آن‌ها صورت نگرفته است، هرچند اهمیت علامت‌ها در تشخیص باعث شد تا در تحقیق انجام شده حداقل از تعداد و مکان آن‌ها استفاده شود. ولی قطعاً درصورتی که بتوان علامت‌ها یعنی نقاط، سرکش و دسته را تشخیص داد، نقش قابل توجهی می‌توانند در بازشناسی زیرکلمات داشته باشند.

۴- **استفاده از اطلاعات متن**: در روش پیشنهادی تمرکز بر روی شناسایی زیرکلمه است، اما بایستی توجه داشت که هر خواننده از خود متن نیز اطلاعات با ارزشی استخراج می‌کند که در این تحقیق این کار انجام نشده است. به طور مثال می‌توان از زیرکلمات قبل و بعد از هر زیرکلمه در شناسایی زیرکلمه استفاده کرد.

فهرست مراجع

[1] Tagougui, N., Kherallah, M., & Alimi, A. M. (2013). Online Arabic handwriting recognition: a survey. *International Journal on Document Analysis and Recognition (IJDAR)*, *16*(3), 209-226.

[۲] برهانی یزدی، ب. (۱۳۸۹) "نقش فناوری OCR در ایجاد کتابخانه دیجیتالی"، *نشریه الکترونیکی سازمان کتابخانه ها*، (۹)۲.

[3] Ziaratban, M., Faez, K., & Faradji, F. (2007, September). Language-based feature extraction using template-matching in Farsi/Arabic handwritten numeral recognition. In *Document Analysis and Recognition, 2007. ICDAR 2007. Ninth International Conference on* (Vol. 1, pp. 297-301). IEEE.

[4] Seni, G., Srihari, R. K., & Nasrabadi, N. (1996). Large vocabulary recognition of on-line handwritten cursive words. *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, *18*(7), 757-762.

[5] Scholey, I. (2006). Digitizer technology Comparison: Inductive and Resisitive. *Pen Computing Magazine.* http://pencomputing.com/features/wacomdigitizercomparison.html. (Last online: 1/22/2015)

[6] Al Hamad, H. A., & Abu Zitar, R. (2010). Development of an efficient neural-based segmentation technique for Arabic handwriting recognition. *Pattern Recognition*, *43*(8), 2773-2798.

[7] Hu, M. K. (1962). Visual pattern recognition by moment invariants. *Information Theory, IRE Transactions on*, *8*(2), 179-187.

[۸] رضوی م. و کبیر ا. (۱۳۸۳). یک پایگاه داده برای دستنوشته برخط فارسی. *ششمین کنفرانس سیستم‌های هوشمند*، ۲۱۸-۲۲۵، کرمان.

[9] Saabni, R., & El-Sana, J. (2009, July). Hierarchical on-line arabic handwriting recognition. In *Document Analysis and Recognition, 2009. ICDAR'09. 10th International Conference on* (pp. 867-871). IEEE.

[۱۰] رضوی س.م و کبیر ا. (۱۳۸۴). روشی ساده برای بازشناسی زیرکلمات فارسی. *نشریه مهندسی برق و مهندسی کامپیوتر ایران*، (۲)۳، صص ۶۳-۷۲.

[۱۱] رضوی س.م و کبیر ا. (۱۳۸۷). بازشناسی برخط کلمات دستنویس فارسی با واژگانی گسترده. *پنجمین کنفرانس ماشین بینایی و پردازش تصویر*.

[12] Faradji, F., Faez, K., & Mousavi, M. H. (2007, November). An HMM-based online recognition system for farsi handwritten words. In *Intelligent and Advanced Systems, 2007. ICIAS 2007. International Conference on* (pp. 1187-1192). IEEE.

[13] Faradji, F., Faez, K., & Nosrati, M. S. (2007, November). Online Farsi handwritten words recognition using a combination of 3 cascaded RBF neural networks. In *Intelligent and Advanced Systems, 2007. ICIAS 2007. International Conference on* (pp. 134-138). IEEE.

[14] Daifallah, K., Zarka, N., & Jamous, H. (2009, July). Recognition-based segmentation algorithm for on-line arabic handwriting. In *Document Analysis and Recognition, 2009. ICDAR'09. 10th International Conference on* (pp. 886-890). IEEE.

[15] Potrus, M. Y., Ngah, U. K., & Sakim, H. A. M. (2010, December). An effective segmentation method for single stroke online cursive Arabic words. In *Computer Applications and Industrial Electronics (ICCAIE), 2010 International Conference on* (pp. 217-221). IEEE.

[16] Eraqi, H. M., & Azeem, S. A. (2011, September). An On-line Arabic Handwriting Recognition System: Based on a New On-line Graphemes Segmentation Technique. In *Document Analysis and Recognition (ICDAR), 2011 International Conference on* (pp. 409-413). IEEE.

[17] El Abed, H., Margner, V., Kherallah, M., & Alimi, A. M. (2009, July). Icdar 2009 online arabic handwriting recognition competition. In *Document Analysis and Recognition, 2009. ICDAR'09. 10th International Conference on* (pp. 1388-1392). IEEE.

[18] Ghods V. & Kabir E. (2010) .Feature Extraction for Online Farsi Characters. 12th *International Conference on Frontiers* in Handwriting Recognition, (pp. 477-482), Bari, Italy.

[۱۹] سلیمانی باغشاه م. باقری شورکی س. و کسائی ش . (۱۳۸۵). بازشناسی و یادگیری حروف مجزای برخط فارسی به روش فازی. *چهاردهمین کنفرانس مهندسی برق ایران*. تهران.

[۲۰] رضوی س.م. و کبیر ا. (۱۳۸۴). روشی ساده برای بازشناسی برخط حروف مجزای فارسی. *نشریه دانشکده مهندسی دانشگاه فردوسی مشهد*. سال هفدهم، شماره دوم، صص ۶۳-۷۲

[۲۱] فرکی م. و پالهنگ م. (۱۳۸۹). بازشناسی برخط حروف فارسی بر پایه مدل مخفی مارکوف. *مجله مهندسی برق دانشگاه تبریز*، شماره ۱، ج ۴۰: صص ۲۳-۳۴.

[22] Douglas, D. H., & Peucker, T. K. (1973). Algorithms for the reduction of the number of points required to represent a digitized line or its caricature. *Cartographica: The International Journal for Geographic Information and Geovisualization*, *10*(2), 112-122.

[23] Biadsy, F., El-Sana, J., & Habash, N. Y. (2006). Online arabic handwriting recognition using hidden markov models. Proceedings of the 10th International Workshop on Frontiers of Handwriting and Recognition.

[24] Kherallah, M., Bouri, F., & Alimi, A. M. (2009). On-line Arabic handwriting recognition system based on visual encoding and genetic algorithm. *Engineering Applications of Artificial Intelligence*, *22*(1), 153-170.

[25] Al-Habian, G., & Assaleh, K. (2007, November). Online Arabic handwriting recognition using continuous Gaussian mixture HMMs. In *Intelligent and Advanced Systems, 2007. ICIAS 2007. International Conference on* (pp. 1183-1186). IEEE.

[۲۶] حلاوتی ر. (۱۳۸۸). استخراج خودکار مفاهیم با هدف افزایش کارایی بازشناسی الگوهای ترتیبی. پایان نامه دکتری، دانشگاه صنعتی شریف، تهران.

[27] Halavati, R., & Shouraki, S. B. (2007). Recognition of Persian online handwriting using elastic fuzzy pattern recognition. *International Journal of Pattern Recognition and Artificial Intelligence*, *21*(03), 491-513.

[28] Theodoridis, S., & Koutroumbas, K. (2008). Pattern recognition. *IEEE TRANSACTIONS ON NEURAL NETWORKS*, *19*(2), 376.

[29] Wagner, R. A., & Fischer, M. J. (1974). The string-to-string correction problem. *Journal of the ACM (JACM)*, *21*(1), 168-173.

[۳۰] تقی زاده، ه. (بهمن ۱۳۹۱). Online Persian Handwriting Recognition. پایان نامه ارشد، دانشگاه شاهرود، شاهرود.

[۳۱] قدس و . و کبیر ا. (۱۳۹۰). بررسی شیوه های متداول دستنوشته های برخط فارسی به منظور استفاده در بازشناسی آن‌ها. *مجله مهندسی برق دانشگاه تبریز، شماره ۱، ج ۴۱: صص ۲۲-۳۲*

Abstract

In this study, a method for online Persian handwritten recognition, based on the recognition of sub-words' composing letters is proposed. The input of systems for online recognition is a set of points related to the handwritten which are generally gathered by handheld digital systems like tablets and smart phone at the time user writes on the screen by touch or electronic pen. So this is called online.

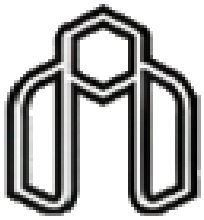
In proposed method in a simplification step, the input data are represented as a set of horizontal and vertical lines with two direction and magnitude vector that these vectors are actually the extracted features of the data. Then reference patterns related to alphabets on various parts of subwords are extracted from feature vectors to create the knowledge base.

In the recognition step, a list of subwords are suggested containing the composing letters recognized by template matching technique with a proposed dynamic programming algorithm. And finally in post-processing step, we filter the suggestions that are not available in dictionary or are not in agreement with the input data in terms of number and or position of sub-strokes.

by the simplification and the proposed algorithm to calculate the edit distance, The success percent of recognition of Persian isolate letters is 95.2% and the percent for the sub words is 63.29% in the first three suggested subwords.

Although the success percent for this method is lower than the holistic approaches but the proposed method has a main advantage over the holistic methods in recognition of sub-words composing letters. To recognize the new sub-words by the proposed method, we just need to add them to the system's dictionary, but for hoslistic methods we require new samples of handwritten.

Keywords: Online Persian Handwritten Recognition, Template matching, Edit distance



Shahrood University of Technology

Faculty of Electrical Engineering and Robotics

An Analytical Approach for On-Line Persian Handwritten Recognition

Alireza Fakhri Nooshabadi

Supervisor:

Dr. A. Ahmadi fard

Advisor:

Dr. H.Khosravi

Date: Feb 2015