باسمه‌تعالی

بینایی ماشین - گزارش ۵

استخراج ویژگی

تهیه و تنظیم:

بهاره بافنده مایوان-۹۳۱۶۴۸۱۱۲۷

موعد تحویل: ۹۴/۳/۱۱

تاریخ تحویل: ۹۴/۳/۱۱

چکیده:

OCR عبارت است از بازشناسی خودکار متون موجود در تصاویر اسناد و تبدیل آنها به متون قابل جستجو و ویرایش توسط رایانه. یک سیستم OCR از بخشهای متعددی تشکیل شده است. ابتدا باید تصویر ورودی آنالیز شده و اگر متن آن دارای چرخش است، اصلاح شود. پس از اصلاح چرخش باید موقعیت بلوکهای متنی، شکل و جدول در تصویر سند مشخص گردد. پس از تعیین موقعیت بلوکهای مختلف (ناحیه بندی یا آنالیز ساختار سند)، باید بلوکهای متنی بازشناسی شوند؛ یعنی خطوط متنی پیدا شده و سپس موقعیت کلمات مشخص شود و در مرحله بعد، موقعیت حروف در کلمه مشخص خواهد شد، در نهایت تک تک حروف باید شناخته شده و با یکدیگر ترکیب شوند تا کلمه‌ی معادل آنها مشخص گردد. در این مینی‌پروژه بر آن هستیم تا یک سیستم OCR برای شناسایی ارقام تایپی فارسی طراحی کنیم. برای این منظور دو روش استفاده از ثابت‌های گشتاور و توصیفگر فوریه را برای استخراج ویژگی بکار گرفته و برای کلاس‌بندی نیز از فاصله‌ی بردار استخراج شده برای هر رقم با میانگین بردار ویژگی نمونه‌های آموزشی در هر کلاس استفاده می‌کنیم.

شرح تکنیکی پروژه:

۱. مقدمه

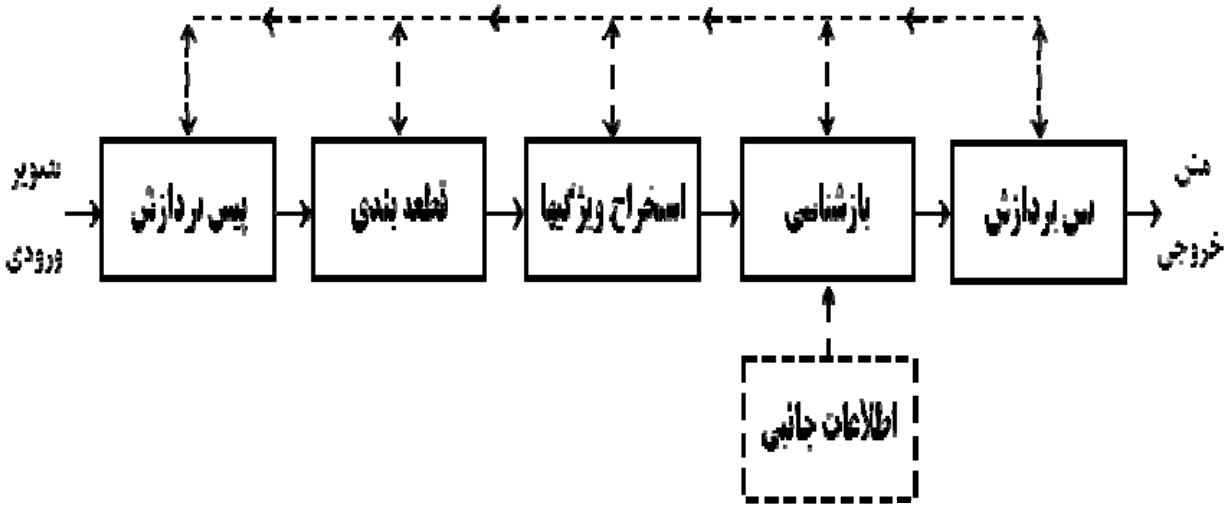
در چند دهه گذشته بازشناسی الگوهای نوشتاری شامل حروف، ارقام و دیگر نمادهای متداول در اسناد نوشته شده به زبان های مختلف، توسط گروه‌های مختلفی از محققین مورد مطالعه و بررسی قرار گرفته‌است. نتیجه این تحقیقات منجر به پیدایش مجموعه‌ای از روش‌های سریع و تا حد زیادی مطمئن، موسوم به OCR یا بازشناسی نوری حروف شده‌است. اصطلاح OCR به تکنیک های اطلاق می شود که در تصاویر اسکن یا فکس‌شده، نواحی متنی را تشخیص می‌دهند سپس این نواحی (تصویری) را به متن قابل ویرایش تبدیل می‌نمایند. استفاده از سیستم های OCR دو مزیت عمده دارد:

الف) افزایش چشمگیر سرعت دسترسی به اطلاعات؛ زیرا در متن برخلاف تصویر، امکان جستجو و ویرایش وجود دارد.

ب) کاهش فضای ذخیره سازی؛ زیرا حجم فایل متنی استخراج شده از یک تصویر، معمولا بسیار کمتر از حجم خود فایل تصویری است.

۲. معرفی بخش‌های مختلف یک سیستم OCR

شکل (الف) نمودار بلوکی یک سیستم OCR را نمایش می‌دهد. لازم به ذکر است که سیستم‌های مختلف با توجه به الگوریتم کلی به‌کار رفته و سطح انتظارات از عملکرد نرم‌افزار، ممکن است فاقد یک یا چند مرحله از مراحل نمایش داده شده باشند.



شکل (الف) نمودار بلوکی یک سیستم OCR

۲. ۱. پیش‌پردازش

این مرحله شامل کلیه‌ی اعمالی می‌شود که بر روی سیگنال تصویری خام صورت می‌گیرند تا موجب تسهیل روند اجرای فازهای بعدی مانند باینری کردن تصویر، حذف نویز، هموارسازی، نازک‌سازی، تشخیص زبان و فونت کلمات و.... گردند. مهمترین گام‌های پیش‌پردازش عبارتند از:

۱- کاهش نویز.

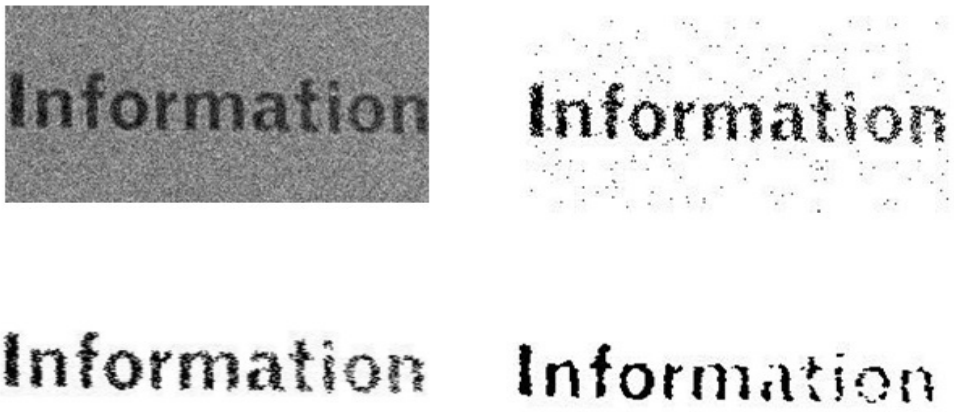
۲- نرمالیزه نمودن داده ها.

۳- فشرده سازی

۲. ۱. ۱. کاهش نویز

نویز ایجاد شده به‌واسطه‌ی ابزارهای نگارشی یا دستگاه‌های اسکنر نوری می‌تواند منجر به ایجاد نقیصه‌هایی مانند تولید قطعه خط‌های گسسته، اتصال بین خطوط، ایجاد فضاهای خالی در خطوط متن، پر شدن حفره‌های موجود در تصویر برخی حروف و ... گردد. همچنین اعوجاج‌های مختلف شامل تغییرات محلی، منحنی شدن گوشه‌های

حروف، تغییر شکل و یا خوردگی حروف را نیز بایستی مد نظر قرار داد. در شکل (ب) نمونه‌هایی از نویز ایجاد شده در تصویر را به نمایش گذاشته‌ایم. قبل از مرحله‌ی بازشناسی حروف لازم است که این نقایص برطرف شوند. یکی از تکنیک‌های مختلف کاهش نویز فیلتر کردن می‌باشد. این روش به حذف نویز کمک می‌کند و ناصافی‌های بدنه‌ی حروف را که معمولا به‌وسیله‌ی سطوح نگارش ناهموار (در متون دست‌نویس) و یا نرخ نمونه‌برداری ضعیف دستگاه-های اخذ داده ایجاد می‌شوند، کاهش می‌دهد. فیلترهای حوزه‌ی مکانی و یا فرکانسی متعددی را می‌توان برای این منظور طراحی کرد. ایده‌ی اصلی در این روش کانوالو کردن یک ماسک از پیش تعریف شده با تصویر جهت تخصیص یک مقدار جدید به پیکسل برحسب تابعی از مقادیر پیکسل‌های مجاور است. فیلترها را می‌توان برای مقاصد مختلفی چون هموارسازی، شارپ کردن، اعمال سطوح آستانه، حذف پس زمینه‌ی بافت‌گونه یا رنگی خفیف و تنظیم کنتراست طراحی نمود.

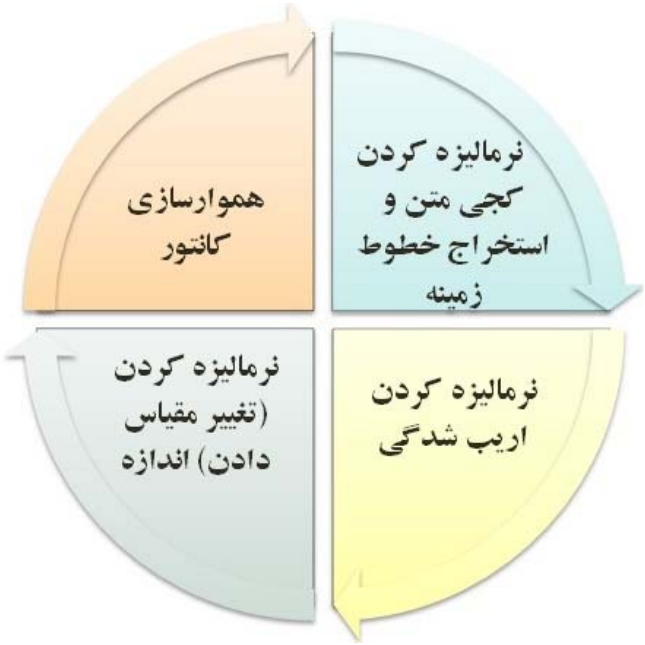


شکل (ب) نمونه‌ای از تصاویر نویزی در ورودی سیستم OCR

۲. ۱. ۲. نرمالیزه کردن داده‌ها

روش های نرمالیزه کردن داده ها به حذف تغییرات نگارشی کمک می‌کند و داده‌های استاندارد شده‌ای را

نتیجه می‌دهد. در شکل (ج) روش‌های پایه نرمالیزه‌کردن نشان داده‌شده‌است. این روش‌ها عبارتند از: ۱-نرمالیزه-کردن کجی متن و استخراج خطوط زمینه، ۲-نرمالیزه‌کردن اریب‌شدگی، ۳-نرمالیزه‌کردن اندازه (تغییر مقیاس دادن) و ۴-هموارسازی کانتور.



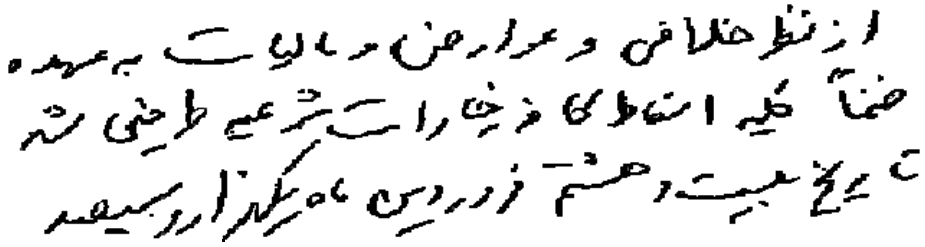
شکل (ج) روش‌های پایه نرمالیزه‌کردن

۲. ۱. ۲. ۱. نرمالیزه کردن کجی متن و استخراج خطوط زمینه

به دلیل بی دقتی در مرحله اسکن یا بی دقتی نویسنده در هنگام نگارش متن دست نوشت، ممکن است خطوط متن نسبت به تصویر، اندکی انحراف یا چرخش داشته باشند. شکل (د) نمونه‌ای از این وضعیت را نشان می‌دهد. این وضع ممکن است کارآیی الگوریتم به کار رفته در طبقات بعدی سیستم OCR را تحت تاثیر قرار دهد؛ چرا که یکی از مفروضات در بیشتر روش های قطعه بندی، کج نبودن تصویر متن ورودی است و در نتیجه لازم که نقیصه، آشکار و تصحیح گردد. آشکارسازی خط زمینه در بسیاری از تکنیک های قطعه بندی و بازشناسی متون فارسی عربی لاتین، نقش اساسی دارد. علاوه بر این، برخی از کاراکترها مانند « ۹ » و « q » در نگارش لاتین و « ۰ »

(نقطه) و «0 » و (صفر) در نگارش فارسی را به واسطه موقعیت نسبی شان نسبت به خط زمینه می‌توان آشکار ساخت.

کلیه الگوریتم‌های توسعه داده شده برای آشکارسازی کجی صفحه، بروی صفحات متنی با ترازبندی یکنواخت، دقیق عمل می‌کنند. الگوریتمی کار آمدتر است که به واسطه حضور مواردی نظیر گرافیک، پاراگراف‌های دارای کجی متفاوت، اعوجاج های منحنی - خطی ظاهر شونده در کتاب های فتوکپی شده، نواحی وسیع پیکسل‌های سیاه نزدیک حاشیه صفحه و خطوط متنی منحصر و کوتاه، دقت آن کمتر دستخوش تغییر شود.



شکل (ج) تصویری از یک متن که کج اسکن شده‌است

روش های بکار رفته برای تصحیح کجی خطوط زمینه در متون لاتین را می توان به چهار گروه اصلی دسته بندی کرد که عبارت اند از:

۱. به کارگیری هیستوگرام تصویر

۲. استفاده از روش خوشه بندی نزدیکترین همسایه ها

۳. روش همبستگی متقابل بین حروف

۴. تبدیل هاف

اغلب پس از آشکارسازی کجی، تصویر صفحه در جهت اصلی چرخانده می شود تا در عملیات تحلیل قالب بندی متن و OCR با سهولت و دقت بیشتری انجام پذیرد. نمونه‌برداری مجدد مورد نیاز برای این منظور که باید بروی

صفحات باینری شده اعمال گردد، ممکن است الگو کاراکترها را تغییر دهد. در این حالت به جای چرخاندن تصویر می توان الگوریتم های پردازشی را به نحوی اصلاح نمود که اثر چرخش در آنها لحاظ گردد. همچنین می‌توان تصویر سند را قبل از باینری کردن، تغییر داد یا اینکه مقدار چرخش را از روی انتقال های کوچک و بدون اعوجاج کل بلوک های متنی، تقریب زد.

۲. ۱. ۲. ۲. نرمالیزه کردن اریب شدگی

در متون چاپی فارسی و لاتین، کاراکترهای دارای قالب ایتالیک از راستای عمود، انحراف دارند. در متون دست نوشت نیز برخی نویسنده‌ها حروف را به صورت زاویه دار می نویسند. این پدیده به عنوان اریب شدگی شناخته می شود و ممکن است دقت برخی الگوریتم های قطعه بندی یا بازشناسی را تحت تاثیر قرار دهد و از این‌رو در این سیستم ها لازم است که در مرحله پیش پردازش، میزان اریب بودن کاراکترها شناسائی و تصحیح گردد. اریب شدگی به‌صورت زاویه‌ی شیب بین طویلترین زیرحرف در یک کلمه و جهت عمودی تعریف می‌شود. نرمالیزه‌کردن اریب، به منظور نرمالیزه‌نمودن کلیه‌ی کاراکترها به یک فرم استاندارد بکار می‌رود. معمول‌ترین روش در تخمین میزان اریب‌شدگی، محاسبه‌ی زاویه‌ی متوسط اجزاء نزدیک به خط عمود است. در استخراج خطوط عمودی از کاراکترها به‌وسیله‌ی دنبال کردن مؤلفه‌های کد زنجیره‌ای توسط یک جفت فیلتر یک بُعدی انجام می‌پذیرد. مختصات شروع و پایان هر خط، زاویه‌ی اریب را بدست می‌دهد.

۲. ۱. ۲. ۳. نرمالیزه کردن اندازه

در سیستم های OCR اغلب تصاویر کلمات خیلی کوچک یا خیلی بزرگ، به یک اندازهٔ استاندارد نرمالیزه می‌شوند. تا بدین ترتیب عملیات بازشناسی، مستقل از اندازه فونت متن گردد. این عمل معمولا با نمونه برداری مجدد تصویر انجام می‌گیرد. روش‌های بازشناسی حروف ممکن است نرمالیزه کردن اندازه را در دو جهت افقی و عمودی انجام دهند. هر کاراکتر به تعدادی ناحیه تقسیم می‌شود و هریک از این نواحی به‌صورت جداگانه تغییر مقیاس

داده می‌شوند. روش هایی نظیر Bilinear یا Bicubic بر روی تصاویر سطح خاکستری به نحو مناسبی عمل می‌کنند.

۲. ۱. ۲. ۴. هموارسازی کانتور

خط تشکیل دهنده مرز یک کاراکتر را کانتور آن کاراکتر گویند. در متون دست نوشت، به واسطه لرزش یا حرکت ناخواسته دست نویسنده در هنگام نگارش، ممکن است که کانتور حروف ناصاف شود. این وضع در سیستم های بازشناسی حروف چاپی و دست نوشت نیز، به دلیل تغییر مقیاس حروف یا وجود نویز در مرحله اسکن تصاویر ممکن است ظاهر گردد. روش های هموارسازی کانتور، به منظور جبران این نقیصه مورد استفاده قرار می گیرند. به طور کلی هموارسازی کانتور، تعداد نقاط نمونه مورد نیاز برای نمایش کاراکتر را کاهش می‌دهد و در نتیجه کارآیی مراحل پردازشی باقیمانده را بهبود می‌بخشد.

۲. ۱. ۳. فشرده‌سازی

این نکته پذیرفته شده‌است که تکنیک‌های کلاسیک فشرده‌سازی تصاویر که تصاویر را از حوزه مکانی به حوزه‌های دیگر منتقل می‌کند، برای بازشناسی حروف مناسب نیستند. در بازشناسی حروف، عمل فشرده‌سازی نیازمند آن دسته از تکنیک‌های حوزه مکانی است که اطلاعات شکلی را حفظ می‌نمایند. دو تکنیک متعارف فشرده سازی، یکی تکنیک اعمال سطح آستانه (به منظور باینری یا دو سطحی کردن تصاویر سطح خاکستری) و دیگری نازک سازی می‌باشند.

الف- باینری (دو سطحی) کردن تصاویر متن: تصاویر دیجیتال به یکی از سه صورت تصاویر رنگی، تصاویر خاکستری (مشابه تصویر یک تلویزیون سیاه و سفید که رنگ تصویر به صورت سیاه و سفید و طیفی از رنگ های خاکستری ظاهر می‌شود) و تصاویر دوگانی یا دو سطحی (مشابه تصویر یک سند فکس‌شده که رنگ پیکسل های تصویر، تنها سیاه یا سفید است) می‌باشند. به منظور کاهش حجم ذخیره‌سازی مورد نیاز و افزایش سرعت و

سهولت پردازش، اغلب مطلوب است که با انتخاب یک سطح آستانه، تصاویر سطح خاکستری یا رنگی را به تصاویر باینری تبدیل نمود.

ب- نازک سازی: با این عمل تصویر کاراکترها به تصویری با عرض یک پیکسل تبدیل می شود؛ درست مثل اینکه کاراکترها با یک قلم نوک باریک نوشته شده باشند. نازک سازی در حالی که کاهش قابل ملاحظه‌ای در حجم داده ها ایجاد می کند، اطلاعات شکل کاراکتری را نیز حفظ می نماید. شکل (د) نمونه‌ای از اعمال عملیات نازک‌سازی را بر روی یک تصویر متنی نشان می‌دهد.

شکل (د) اعمال عملیات نازک‌سازی روی یک تصویر متنی

دو روش پایه برای نازک سازی عبارتند از :

- نازک سازی از طریق پیکسل
- نازک سازی غیر از طریق پیکسل

نازک سازی از طریق پیکسل بصورت محلی و تکراری تصویر را مورد پردازش قرار می دهد تا وقتی که از تصویر کاراکتر تنها اسکلت آن به عرض یک پیکسل باقی بماند. این روش نسبت به نویز بسیار حساس بوده، ممکن است تصویر کاراکتر را مخدوش سازد. از سوی دیگر، روش‌های نازک سازی غیر از طریق پیکسل، طی فرایند نازک سازی مقداری از اطلاعات سراسری دربارهٔ کاراکتر را مورد استفاده قرار می دهند. این روشها یک خط مرکزی یا میانهٔ بخصوص از تصویر پرتر را بدون آزمایش همهٔ پیکسلها تولید می نمایند.

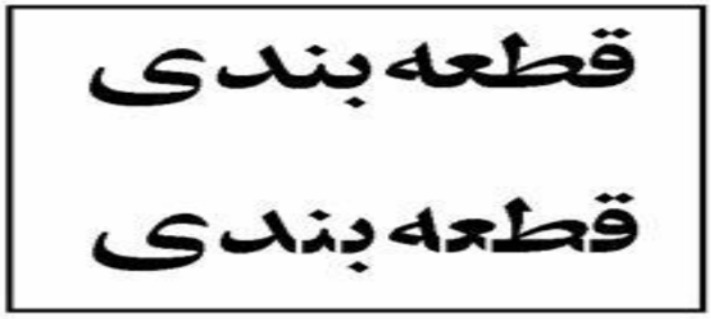
ج- بازشناسی خط، زبان و فونت: بازشناسی خط، تعداد کلاس‌های مختلف نمادهایی را که باید مورد ملاحظه قرار گیرند، کاهش می دهد. شناسائی زبان متن، به منظور بکارگیری مدل‌های متنی خاص ضرورت دارد. طبقه بندی فونت‌ها، کاهش تعداد شکل‌های مختلف حروف در هر کلاس که لازم است در فرایند بازشناسی لحاظ گردند را به دنبال دارد و سبب می‌شود که امر شناسائی، تنها به یک کلاس فونت محدود گردد. بازشناسی خط و زبان و فونت در کاربردهایی مانند نمایه سازی و دستکاری اسناد نیز مطلوب می باشد.

۲. ۲. قطعه‌بندی

قطعه‌بندی مرحله‌های بسیار مهم برای سیستم‌های OCR مخصوصا OCR فارسی و عربی (که حروف کلمات به صورت سرهم می‌شوند) می‌باشد. قطعه بندی به دو نوع گونه تقسیم می‌شود.

۱. قطعه‌بندی بیرونی عبارت است از تفکیک قسمت‌های مختلف تصویر نظیر متن، گرافیک و خطوط و نیز جدا کردن بخش های مختلف متن مانند پاراگراف ها، جملات یا کلمات؛

۲. قطعه‌بندی درونی که منظور از آن، جداسازی حروف کلمات مخصوصا در مورد کلمات سرهم نوشته شده در متون لاتین، یا در رسم الخطهای پیوسته نظیر فارسی و عربی است، شکل (ه) این موضوع را نشان می‌دهد. همچنین حروفی که در متن اصلی جدا بوده اند، اما به خاطر کیفیت پایین دستگاه اسکنر به هم چسبیده‌اند، توسط این دسته از تکنیک ها از یکدیگر جدا می‌شوند.



شکل (ه) قطعه‌بندی یک کلمه به حروف

مرحله قطعه بندی بیرونی، بحرانی‌ترین و حساس‌ترین قسمت در حوزه تحلیل تصویر اسناد می‌باشد و یک مرحله ضروری برای سیستم های OCR برون خط محسوب می‌شود. گرچه مبحث تحلیل اسناد با روش‌ها و تکنیک‌های خاص خود یک حوزه تحقیقاتی تا حدی متفاوت نسبت به OCR است. اما تقسیم بندی تصویر سند به نواحی متنی و غیر متنی، یک بخش لاینفک در حوزه OCR یک بخش لاینفک در نرم افزارهای OCR به حساب می آید. بنابراین برای افرادی که در حوزه OCR تحقیق می‌کنند، داشتن دانش عمومی از تکنیک های آنالیز اسناد ضرورت دارد.

تحلیل پیکربندی صفحات در سه مرحله صورت می‌گیرد: مرحله اول تفکیک نواحی متنی در تصویر از نواحی غیر متنی (شامل گرافیک و خطوط) است. مرحله دوم تحلیل ساختاری است که با قطعه بندی نواحی متنی به بلوک هایی از تصویر سند (نظیر پاراگراف ها، ردیف، کلمه، و ...) مرتبط می باشد. مرحله سوم که تحلیل عملکردی نام دارد، با استفاده از اطلاعت مکانی، اندازه و قوانین مختلف صفحه بندی، عملکرد هر یک از اجزای سند (نظیر عنوان، چکیده و ...) را تعیین می نماید.

نقطه تمایز اصلی میان سیستم های فارسی OCR فارسی و لاتین برای متون چاپی، در مرحله قطعه بندی درونی نهفته است؛ چرا که حروف کلمات در نگارش فارسی برخلاف نگارش رسمی لاتین به صورت سرهم نوشته می شوند و در نتیجه ضرورت انجام صحیح این مرحله برای متون فارسی و عربی نسبت به متون لاتین اهمیت فوق العاده بیشتری دارد. با وجود فعالیت نسبتا چشمگیر گذشته و تنوع تکنیک‌های عرضه شده، قطعه بندی متون پیوسته (بخصوص متون دست نوشته پیوسته) به حروف، هنوز هم یک مسئله قابل بررسی مانده است. روش های قطعه بندی حروف به سه دسته تقسیم می شوند:

۱. قطعه بندی صریح

۲. قطعه بندی ضمنی

۳. تکنیک های ادغام شده

در مواردی همچون متون فارسی که حروف به صورت سرهم نوشته می شوند، سه رویکرد مختلف در بازشناسی برون خط متون کلمات یا زیر کلمات وجود دارد:

۱. رویکرد مبتنی بر قطعه بندی کلمات

۲. رویکرد مبتنی بر بازشناسی کلمه به عنوان یک الگوی واحد

۳. رویکرد ترکیبی

در رویکرد بازشناسی مبتنی بر قطعه‌بندی، ابتدا کلمه در مرحله جداسازی به حروف یا زیرحروف، شکسته می‌شود؛ آنگاه قطعات جدا شده بازشناسی می‌شوند و از کنار هم قرار گرفتن آنها، کلمه شناسائی خواهدشد. روش‌های به کار گرفته شده در این رویکرد به دو گروه مختلف تقسیم می‌شود: ۱- تقطیع کلمه به حروف ۲- تقطیع کلمه به زیرحروف. در گروه اول کلمه به حروف جداسازی می شوند و با شناسائی حروف جدا شده، کلمه بازشناسی می گردد. در گروه دوم کلمه به زیر حروف مثل پاره منحنی ها و ساختار های پایه دیگر جداسازی می شود و با شناسائی زیر حروف و ترکیب آنها، کلمه بازشناسی می گردد. در این رویکرد نمی توان در ابتدا مرز حروف را به طور کامل مشخص کرد، بلکه حروف به ترتیب از ابتدا به انتها کلمه، بازشناسی و جداسازی می‌شوند. در هیچ یک از دو رویکرد نخست که مبتنی بر جداسازی هستند، به شکل کلی کلمه توجه نمی‌شود و سعی بر آن است که با بازشناسی حروف یک کلمه، آن کلمه شناخته شود.

در رویکرد بازشناسی کلمه به عنوان یک الگوی واحد، تلاشی برای تقطیع کلمه به حروف موجود در کلمه صورت نمی‌گیرد و کلمه در قالب یک الگو بررسی می‌گردد.

قطعه بندی غلط کاراکترها، عامل بسیاری از خطاهای OCR است (مانند m→ rn، rn→m، ز→ر، ز→ژ، ت→ث). میزان دقت یک الگوریتم قطعه بندی به سبک نگارش حروف، کیفیت دستگاه چاپ(کاراکتر های ایتالیک لکه دار برای قطعه بندی دارای اشکال می باشند)، و نیز نسبت اندازه فونت به قدرت دستگاه اسکنر (تابع گسترش

نقاط و نرخ نمونه برداری مکانی) بستگی دارد. نتیجه مطلوب مرحله قطعه بندی، تصویری است که تنها حاوی یک کاراکتر باشد و بجز پیکسل های پس‌زمینه، هیچ شی دیگری را شامل نشود. اما هنگامی‌که که اشیای چاپی، در تصویر ورودی خیلی نزدیک به هم ظاهر شوند (مانند نقشه‌های هیدروگرافیکی)، این منظور همواره قابل حصول نخواهد بود. غالبا در چنین حالتی دیگر اشیای یا کاراکترهای چاپی، به طور تصادفی در داخل تصویر کاراکتر قرار می گیرند و احتمالا ویژگی های استخراج شده را تحریف می نمایند. این مورد یکی از دلایلی است که بیان می‌دارد چرا هر سیستم بازشناسی حروف، یک گزینه وازدگی دارد.

۲. ۳. بازنمایی (استخراج ویژگی‌ها)

این مرحله یکی از مراحل بسیار با اهمیت در سیستم‌های OCR است؛ چرا که نتایج حاصل از این مرحله، مستقیما بروی کیفیت مرحله بازشناسی اثر می گذارد. در مرحله بازنمایی، به هر الگوی ورودی (کاراکتر یا کلمه بر حسب آن‌که رویکرد سیستم، مبتنی بر کدام یک از دو گروه « قطعه بندی کلمات» یا « شناسائی کلمه به عنوان یک الگوی واحد» باشد)، یک کد یا بردار ویژگی نسبت داده می‌شود که معرف آن الگو در فضای ویژگی‌ها است و آن را از دیگر الگوها متمایز می‌سازد. در انتخاب بردارهای ویژگی لازم است موارد زیر مورد توجه قرار گیرند:

۱. بردار ویژگی هر الگو باید تا حد امکان از بردار ویژگی دیگر الگوها متمایز باشد (فاصله بین بردارهای ویژگی در فضای ویژگی ها، حداکثر باشد).

۲. بردار ویژگی الگوها باید تا بیشترین حد ممکن، خصوصیات شکل و ساختار الگوها را از تصویر آنها استخراج نماید.

۳. تا حد امکان نسبت به نویز، تغییر اندازه و نوع فونت، چرخش، و دیگر تغییرات احتمالی الگوها دارای ثبات باشد.

۴. شرایط ، نوع و خصوصیات الگوهای ورودی در انتخاب بردارهای ویژگی اثر می‌گذارند. به عنوان مثال، باید تعیین نمود که آیا حروف یا کلمات که می بایست تشخیصداده شوند جهت و اندازه مشخصی دارند یا خیر، دست نوشت

هستند یا چاپی، یا اینکه تا چه حد بوسیله نویز، مغشوش شده‌اند. همچنین گاهی کفایت می‌کند که سیستم تنها جوابگوی گروه محدودی از الگوها (مثلا الگوهای با اندازه یا نوع فونت از پیش مشخص‌شده) باشد.

۵. در مورد حروفی که به چندین شکل نوشته می شوند، لازم است که بیش از یک کلاس الگو به یک کاراکتر خاص تعلق یابد.

همانطور که عنوان شد، بازنمایی یک مرحله بسیار مهم در حصول راندمان مناسب برای سیستم بازشناسی حروف است؛ ولی برای دستیابی به عملکرد بهینه، لازم است که دیگر مراحل نیز بهینه گردند و باید توجه نمود که این مراحل، مستقل از هم نیستند. یک روش خاص استخراج ویژگی‌ها، طبیعت خروجی مرحله پیش پردازش را به ما دیکته می کند یا حداقل ما را در انتخابمان محدود می سازد.

مراحل قطعه بندی و بازشناسی، دو وجه تمایز عمده میان سیستم های OCR فارسی و لاتین می باشند. به واسطه وجود تفاوت های اساسی بین نحوه نگارش فارسی و لاتین، امکان اعمال مستقیم تکنیک‌های قطعه‌بندی و بازشناسی مربوط به سیستم های OCR لاتین، برای متون فارسی وجود ندارد. پیچیدگی های مختص نگارش فارسی، بر پیچیدگی‌های الگوریتم‌های این دو مرحله می افزاید. درست به همین دلیل است که بیشتر نرم افزارهای OCR تجار لاتین، قادر به پشتیبانی زبان فارسی و عربی نمی باشند.

۲. ۴. طبقه‌بندی یا بازشناسی (با یک یا چند طبقه‌بندی کننده)

این مرحله شامل روش های برای متناظر ساختن هر یک از الگوهای بدست آمده از مرحله استخراج ویژگی‌ها، با یکی از کلاس های فضای الگو مورد بحث است که از طریق کمینه ساختن فاصله بردار ویژگی های هر الگوی ورودی نسبت به یکی از بردارهای مرجع انجام می گیرد. بردارهای مرجع بردارهای هستند که قبلا از نمونه‌های آموزشی اخذ شده‌اند. تکنیک های عرضه شده در این مرحله را می توان در روش های مربوط به چهار گروه عمومی مبحث شناسی الگو، جستجو نمود: ۱-تطابق قالبی، ۲- تکنیک های آماری، ۳- تکنیک های ساختاری، ۴- شبکه های عصبی.

چهار گروه فوق لزوما مستقل یا مجزا از یکدیگر نمی‌باشند و گاهی می‌توان تکنیک‌های یک گروه را در میان تکنیک‌های مربوط به دیگر گروه‌ها یافت.

۲. ۵. بکارگیری اطلاعات جانبی (پس پردازش)

در این مرحله با استفاده از اطلاعات جانبی (نظیر مجموعه لغات معتبر، اطلاعات آماری مربوط به رخداد حروف، اطلاعات دستوری و معنایی) سعی در بهبود نتایج حاصل از مرحله بازشناسی می گردد. تا قبل از این مرحله هیچ گونه « تمیز کردن » اطلاعات معناشناختی در طول مراحل پردازشی مورد استفاده قرار نگرفته بود. مرحله پیش پردازش سعی در تصویر سند به نحو مقتضی دارد و به علت عدم دسترسی به اطلاعات مفهومی، ممکن است باعث حذف برخی از اطلاعات مهم تصویر گردد. فقدان اطلاعات معنایی در مرحله قطعه بندی می تواند به نتایج حادتر و غیر قابل برگشتی بی‌انجامد؛ چرا که خروجی این مرحله، تعیین کننده مرز الگوهای ورودی می‌باشد. بنابراین واضح است که درصورت فراهم شدن اطلاعات معناشناختی، دقت نتایج بازشناسی به نحو چشمگیری افزایش می‌یابد. مروری بر تحقیقات انجام شده در زمینه بازشناسی حروف نشان می دهد که در صورت استفاده از اطلاعات شکلی بدون بکارگیری اطلاعات معناشناختی، افزایش دقت قابل توجهی نخواهیم داشت. در نتیجه، یکپارچه سازی اطلاعات ساده ترین راه برای منظور، استفاده از یک فرهنگ لغات برای اصلاح خطاهای جزئی است. این کار توسط یک برنامه واژه پرداز(با قابلیت خطایاب املایی) که املای کلمات را کنترل و چندین پیشنهاد برای اصلاح املای کلمات نامانوس ارائه می‌کند، انجام می‌گیرد.

مراحل بکارگرفته‌شده در این تمرین

بخش آموزش

در بخش آموزش ویژگی‌های ارقام را با استفاده از دو روشی که در ادامه آورده‌ایم، استخراج خواهیم کرد. سپس برای هر رقم میانگین بردارهای به‌دست آمده را به عنوان بردار ویژگی آن رقم ذخیره خواهیم‌کرد.

ثابت‌های گشتاور

یکی از ویژگی‌هایی که در این مینی پروژه استفاده شده‌است ثابت‌های گشتاور می‌باشند. این ویژگی نسبت به انتقال، تغییر اندازه، دوران و آیینه‌ای شدن استقلال دارد. برای تصویر $f(x,y)$ با ابعاد $M * N$ گشتاور دوبُعدی مرتبه $(p+q)$ به‌صورت زیر تعریف می‌شود:

$$m_{pq} = \sum_{x=0}^{M-1} \sum_{y=0}^{N-1} x^p y^q f(x,y) \qquad p,q = 0,1,2,\dots$$

برای ایجاد استقلال از دوران، از گشتاور مرکزی استفاده می‌شود:

$$\mu_{pq} = \sum_{x=0}^{M-1} \sum_{y=0}^{N-1} (x-\bar{x})^p (y-\bar{y})^q f(x,y)$$

که در آن

$$\bar{x} = \frac{m_{10}}{m_{00}} \quad , \quad \bar{y} = \frac{m_{01}}{m_{00}}$$

برای اضافه کردن استقلال از انتقال و تغییر اندازه از گشتاور مرکزی مقیاس‌شده استفاده می‌کنیم:

$$\eta_{pq} = \frac{\mu_{pq}}{(\mu_{00})^{\gamma}} \quad , \quad \gamma = \frac{p+q}{2} + 1$$

و در نهایت ثابت‌های گشتاور به‌صورت زیر تعریف می‌شوند:

$$\begin{aligned}
\phi_1 &= \eta_{20} + \eta_{02} \\
\phi_2 &= (\eta_{20} - \eta_{02})^2 + 4\eta_{11}^2 \\
\phi_3 &= (\eta_{30} - 3\eta_{12})^2 + (3\eta_{21} - \eta_{03})^2 \\
\phi_4 &= (\eta_{30} - \eta_{12})^2 + (\eta_{21} - \eta_{03})^2 \\
\phi_5 &= (\eta_{30} - 3\eta_{12})(\eta_{30} + \eta_{12})[(\eta_{30} + \eta_{12})^2 - 3(\eta_{21} + \eta_{03})^2] \\
&\quad + (3\eta_{21} - \eta_{03})(\eta_{21} + \eta_{03})[3(\eta_{30} + \eta_{12})^2 - (\eta_{21} + \eta_{03})^2] \\
&\vdots
\end{aligned}$$

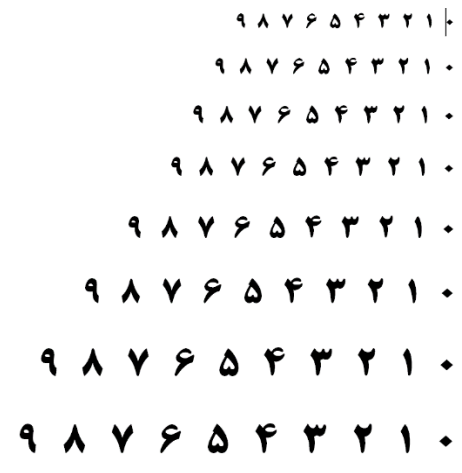
توصیفگر فوریه

برای محاسبه‌ی این ویژگی از یک نقطه مشخص بر روی پیرامون آغاز کرده و مثلا در جهت عقربه های ساعت تمام نقاط پیرامون را به صورت رشته ای از اعداد مختلط بیان می کنیم. از این رشته تبدیل فوریه گرفته و در حوزه فوریه تنها p عدد را حفظ کرده و از بقیه اعداد صرف نظر می کنیم. مجددا این اعداد را با عکس تبدیل فوریه به اعداد مختلط تبدیل می کنیم.

$$S(k) = x_k + jy_k$$

هر چه p کمتر باشد جزئیات بیشتری از بین می رود.

می توان دید که با تعداد کمی از این اعداد باز هم پیرامون تصویر به خوبی قابل بازیابی است. در این پروژه برای استقلال از اسکیل تمامی نمونه های آموزش و تست در مرحله پیش پردازش به سایز یکسانی تبدیل می شوند.



تصویر استفاده شده برای آموزش سیستم

بخش آزمایش

در بخش آزمایش، همانند بخش آموزش ابتدا کاراکترها را جدا کرده و بردار ویژگی هرکدام را استخراج می‌کنیم. سپس این بردار را با بردارهای ویژگی به‌دست آمده در بخش آموزش مقایسه کرده و نزدیکترین آن‌ها را به‌عنوان پاسخ انتخاب می‌کنیم.

تصاویر استفاده شده برای تست سیستم

ارزیابی روش

برای ارزیابی روش‌های انجام‌شده از ماتریس درهم‌ریختگی استفاده می‌کنیم. سطرهای این ماتریس مقادیر واقعی و ستون‌های آن مقادیر پیش‌بینی شده را نشان می‌دهند. بنابراین اعداد روی قطر اصلی ماتریس نشان‌دهنده‌ی تعداد کلاس‌بندی‌های درست خواهند بود.

شرح نتایج:

نتایج مربوط به تصویر ۳۲۲

10	0	0	0	0	0	0	0	0	0
0	10	0	0	0	0	0	0	0	0
0	0	9	0	0	0	0	1	0	0
0	0	0	10	0	0	0	0	0	0
0	0	0	0	9	0	0	1	0	0
0	0	0	0	4	5	1	0	0	0
0	0	0	0	0	0	10	0	0	0
0	0	2	0	1	0	0	2	5	0
0	0	1	0	1	0	0	3	5	0
0	0	0	0	1	0	0	0	0	9

ماتریس درهم ریختگی در روش ثابت گشتاور

10	0	0	0	0	0	0	0	0	0
0	2	0	0	0	0	0	0	2	6
0	0	10	0	0	0	0	0	0	0
0	0	1	2	0	0	0	7	0	0
0	0	0	0	9	0	0	0	0	1
0	0	0	0	0	10	0	0	0	0
0	0	0	0	0	0	8	0	0	2
0	0	0	5	0	0	0	5	0	0
0	0	0	0	0	0	0	1	9	0
0	4	0	0	0	0	1	0	0	5

ماتریس درهم ریختگی در روش توصیفگر فوریه

نتایج مربوط به تصویر ۳۲۳

10	0	0	0	0	0	0	0	0	0
0	10	0	0	0	0	0	0	0	0
0	0	9	0	0	0	0	0	1	0
0	0	0	10	0	0	0	0	0	0
0	0	0	0	8	2	0	0	0	0
0	0	0	0	3	6	1	0	0	0
0	0	0	0	0	0	10	0	0	0
0	0	2	0	0	0	0	3	5	0
0	0	3	0	1	0	0	1	5	0
0	0	0	0	0	0	0	0	0	10

ماتریس درهم ریختگی در روش ثابت گشتاور

9	0	0	0	0	0	0	0	0	1
0	2	0	0	0	0	0	0	3	5
0	0	10	0	0	0	0	0	0	0
0	0	0	2	0	0	0	8	0	0
0	0	0	0	8	0	0	0	0	2
0	0	0	0	0	9	0	0	0	1
0	0	0	0	0	0	7	0	0	3
0	0	0	4	0	0	0	5	1	0
0	0	0	0	0	0	0	2	8	0
0	4	0	0	0	0	1	0	0	5

ماتریس درهم ریختگی در روش توصیفگر فوریه

پیوست:

کد اصلی :

```
clc; clear all; close all;

%%%%%%%%%%%%%%%%%%%%%%%%% Train %%%%%%%%%%%%%%%%%%%%%%%%%%%%%

img = imread('Im321.bmp');
img= imcomplement(im2bw(img,graythresh(img)));
[R,C] = size(img);
lineCnt = 0;
row = 1;
while row < R
    lineCnt = lineCnt + 1;
```

```
    while (sum(img(row,:)) == 0) && (row < R)
        row = row+1;
    end
    temp = img(row,:);
    while (sum(img(row,:)) ~= 0) && (row < R)
        temp = [temp; img(row,:)];
        row = row + 1;
    end
    lines{lineCnt} = temp;
end

MM=zeros(10,3);
MF=zeros(10,1);
for i=1:10
    SM1=0;SM2=0;SM3=0;SF1=0;
    SF4_1=0;SF4_2=0;SF4_3=0;SF4_4=0;
    for l = 1:lineCnt-1
        CC = bwconncomp(lines{l});
        STATS = regionprops(CC,'Image');
        digit = imresize(STATS(i).Image,[65 40]);
        SM1 =SM1+( n(digit, 2, 0) + n(digit, 0, 2));
        SM2 =SM2+((n(digit, 2, 0) + n(digit, 0, 2))^2 + 4*n(digit,1,1)^2);
        SM3 =SM3+((n(digit, 3, 0) - 3*n(digit, 1, 2))^2+(3*n(digit, 2, 1) -
n(digit, 0, 3))^2);
%       SF1 =SF1+(fourier(digit,1));
        f=fourier(digit,4);
        SF4_1 =SF4_1+f(1,1);
        SF4_2 =SF4_2+f(1,2);
        SF4_3 =SF4_3+f(1,3);
        SF4_4 =SF4_4+f(1,4);
    end
    MM(i,1)=SM1/8;
    MM(i,2)=SM2/8;
    MM(i,3)=SM3/8;
%   MF(i,1)=SF1/8;
    MF(i,1)=SF4_1/8;
    MF(i,2)=SF4_2/8;
    MF(i,3)=SF4_3/8;
    MF(i,4)=SF4_4/8;
end

%%%%%%%%%%%%%%%%%%%%%%%% Test %%%%%%%%%%%%%%%%%%%%%%%%%%%%%

img = imread('Im323.jpg');
img= imcomplement(im2bw(img,graythresh(img)));
[R,C] = size(img);
lineCnt = 0;
row = 1;
while row < R
    lineCnt = lineCnt + 1;
    while (sum(img(row,:)) == 0) && (row < R)
        row = row+1;
    end
    temp = img(row,:);
    while (sum(img(row,:)) ~= 0) && (row < R)
        temp = [temp; img(row,:)];
```

```
        row = row + 1;
    end
    lines{lineCnt} = temp;
end
count=1;
TMmnt=zeros(100,3);
TFourier=zeros(100,4);
for l = 1:lineCnt-1
    CC = bwconncomp(lines{l});
    STATS = regionprops(CC,'Image');
    for i=1:10
        digit = imresize(STATS(i).Image,[65 40]);
        TestMmnt(1) = n(digit, 2, 0) + n(digit, 0, 2);
        TestMmnt(2) = (n(digit, 2, 0) + n(digit, 0, 2))^2 + 4*n(digit,1,1)^2;
        TestMmnt(3) = (n(digit, 3, 0) - 3*n(digit, 1, 2))^2+(3*n(digit, 2, 1)
- n(digit, 0, 3))^2;
        TMmnt(count,:)=TestMmnt;

        f=fourier(digit,4);
        SF4(1) =f(1,1);
        SF4(2) =f(1,2);
        SF4(3) =f(1,3);
        SF4(4) =f(1,4);

        TFourier(count,:)=SF4;
        count=count+1;
    end
end

%%%%%%%%%%%%%%%%%%%%%%%% Confusion Matrix %%%%%%%%%%%%%%%%%%%%%%%%%%%%%

M=zeros(10,10);
F=zeros(10,10);
for i=1 : 100
    MinDist1 = 10000;
    MinIndex1 = 1;
    MinDist2 = 10000;
    MinIndex2 = 1;
    for k=10:-1:1
        d1=real(Distance(TMmnt(i,:),MM(k,:)));
        d2=real(Distance(TFourier(i,:),MF(k,:)));
        if d1 < MinDist1
            MinDist1 = d1;
            MinIndex1 = k;
        end
        if d2 < MinDist2
            MinDist2 = d2;
            MinIndex2 = k;
        end
    end
    M(rem(i-1,10)+1, 11-MinIndex1) = M(rem(i-1,10)+1, 11-MinIndex1) + 1;
    F(rem(i-1,10)+1, 11-MinIndex2) = F(rem(i-1,10)+1, 11-MinIndex2) + 1;
end
```

کد تابع Distance:

```
function distance = Distance( A, B )
distance = sqrt(sum((A-B).^2));
end
```

کد تابع fourier:

```
function X = fourier(im,f_num)
im = padarray(im,[1 1]);
brd = bwmorph(im,'remove');
brd = bwmorph(brd,'thin');
st = regionprops(brd,'all');
border = [];
for i = 1:size(st,1)
    border = [border; st(i).PixelList];
end
s = border(:,1) + 1i*border(:,2);
F = fft(s);
F(f_num+1:end) = 0;
X = ifft(F,f_num)';
end
```