دانشگاه شهید بهشتی

دانشکده مهندسی و علوم کامپیوتر

گزارش پروژه پایانی درس یادگیری ماشین

پیاده سازی شناسایی دست نوشته در پایتون
با دو روش مدل مخفی مارکوف و نزدیکترین همسایگان

نگارش

سهیل ضیائی قهنویه

استاد

دکتر احمد علی آبین

تیر ماه ۱۴۰۰

چکیده

شناسایی دست نوشته یکی از تسک های مهم هوش مصنوعی است که در آن انتظار می رود با دریافت تصاویر متون دست نوشته به عنوان ورودی، محتوای نوشته به عنوان خروجی تولید شود. در این پروژه مجموعه داده معروف و پراستفاده اعداد دست نوشته MNIST به عنوان ورودی به دو الگوریتم مدل مخفی مارکوف، و نزدیکترین همسایگان داده شده و میزان دقت آن ها در دسته بندی و تشخیص اعداد اندازه گیری می شود.

کلیدواژه‌ها

شناسایی دست نوشته، مدل مخفی مارکوف، HMM، نزدیکترین همسایگان، KNN، اعداد دست نوشته MNIST، پایتون

فهرست نوشتار

بخش اول: مقدمه

شناسایی دست نوشته[۱]

شناسایی دست نوشته یکی از تسک های مهم هوش مصنوعی است که در آن انتظار می رود با دریافت تصاویر متون دست نوشته به عنوان ورودی، محتوای نوشته به عنوان خروجی تولید شود. چالش های زیادی برای این دسته از مسائل مطرح می شود، مانند متغیر بودن رنگ نوشته، سایز فونت، تفاوت دست خط انسان ها و ... لذا پژوهش بر روی این دسته از مسائل اهمیت زیادی داشته و روش های متعدد یادگیری ماشین از مرسوم ترین راه حل ها برای آن به حساب می آیند.

مدل مخفی مارکوف[۲]

مدل مخفی مارکوف از روش های آماری یادگیری ماشین است که در آن سیستم مدل شده به صورت یک فرایند مارکوف با حالت‌های مشاهده نشده (پنهان) فرض می‌شود و مهمترین کاربردهای آن شامل مسائلی است که در آن هدف پیدا کردن توالی از داده ها از طریق داده های دیگری که به آن وابسته هستند، می باشد.

الگوریتم نزدیک ترین همسایگان[۳]

الگوریتم نزدیک ترین همسایگان از روش های آماری یادگیری ماشین است که برای کلاس بندی و رگرسیون استفاده می شود. در هر دو حالت هدف پیدا کردن کلاس یا مقدار یک نمونه آزمایشی بر اساس نزدیک ترین نمونه های آموزشی در فضای داده ای می باشد.

ساختار گزارش

کامل خواهد شد.

---

۱ Handwriting recognition

۲ Hidden Markov Model (HMM)

۳ K Nearest Neighbors (KNN)

بخش دوم: تشریح مسئله

تعریف مسئله

در قسمت اول این پروژه قرار است شناسایی دست خط بر روی اعداد ۰ تا ۹ از طریق پیاده سازی «مدل مخفی مارکوف» انجام شود. در قسمت دوم، استفاده از الگوریتم یادگیری دیگری مد نظر می باشد و در هر دو مرحله لازم است نتایج ارزیابی و تحلیل گردند.

اهمیت مسئله

این پروژه یک نمونه کوچک از یکی از تسک های مهم هوش مصنوعی به شمار می آید که برای تشخیص متن نوشته شده از طریق پردازش تصویر دست نوشته از منابعی مانند اسناد کاغذی، عکس، صفحه نمایش لمسی و ... به کار می رود.

مجموعه داده مسئله

مجموعه داده ای که به عنوان ورودی در این پروژه استفاده می شود مجموعه داده معروف و پراستفاده اعداد دست نوشته MNIST است که در چهار فایل مجزا (داده های تصاویر آموزشی، داده های برچسب های آموزشی، داده های تصاویر آزمایشی، داده های برچسب های آزمایشی) از وب سایت زیر قابل دریافت است:

http://yann.lecun.com/exdb/mnist/

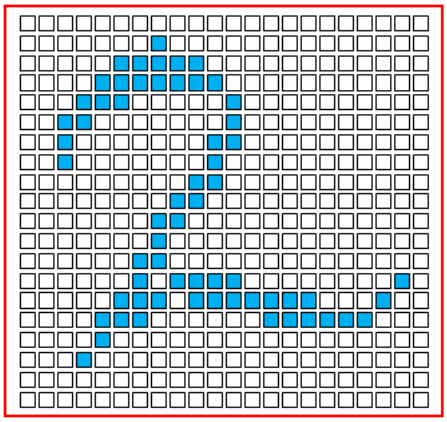
train-images-idx3-ubyte.gz	training set images (9912422 bytes)
train-labels-idx1-ubyte.gz	training set labels (28881 bytes)
t10k-images-idx3-ubyte.gz	test set images (1648877 bytes)
t10k-labels-idx1-ubyte.gz	test set labels (4542 bytes)

بخش سوم: روش پیشنهادی

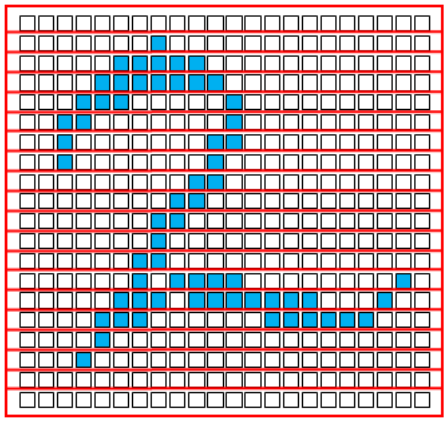
مدل مخفی مارکوف

مدل مخفی مارکوف در اصل از الگوریتم های یادگیری بدون ناظر به شمار می آید، برای استفاده از آن در این مسئله که به نوعی طبقه بندی (classification) است، از ایجاد مدل های مختلف برای هر یک از کلاس ها (اعداد ۰ تا ۹) استفاده خواهد شد. احتمال تولید هر یک از داده های آزمایشی با همه مدل ها محاسبه شده و در نهایت مدلی که بیشترین احتمال را دارد، به عنوان کلاس پیش بینی شده داده آزمایشی انتخاب خواهد شد.

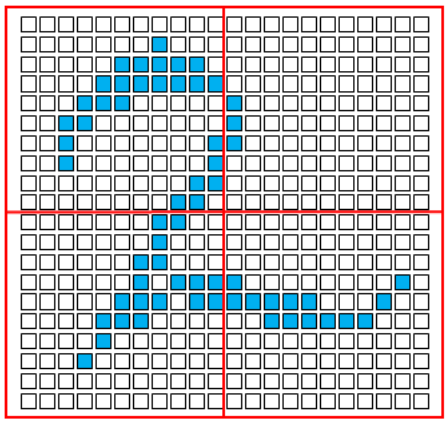
ایده های مختلفی برای نگاشت ویژگی های یک تصویر به وضعیت ها و مشاهدات ارائه شده است. در این پروژه در ابتدا کل پیکسل های تصویر به عنوان دنباله ای از مشاهدات در نظر گرفته شده و به مدل ها آموزش داده می شوند (شکل ۱). در ادامه به منظور بهبود دقت، تصاویر به قسمت هایی تقسیم می شوند و هر قسمت از تصویر به عنوان یک وضعیت به مدل داده می شوند. در مجموع این امکان وجود دارد که با روش های خلاقانه حالت های مختلفی برای تقسیم تصویر و ایجاد تعداد مختلف وضعیت فرض کرد؛ مثلا سطر های تصویر را به عنوان وضعیت های مدل در نظر گرفت (شکل ۲). و یا تقسیم تصویر به چهار قسمت (شکل ۳) که از مزیت شباهت با حرکت واقعی قلم در دست نوشته (بالا به پایین و چپ به راست) بهره می برد.



شکل ۱ ورودی مدل مخفی مارکوف با یک وضعیت



شکل ۲ تقسیم سطر به سطر تصویر، ورودی مدل مخفی مارکوف که به تعداد سطرها وضعیت دارد



شکل ۳ تقسیم تصویر به چهار قسمت، ورودی به مدل مخفی مارکوف با ۴ وضعیت

الگوریتم نزدیک ترین همسایگان

از مزایای الگوریتم نزدیک ترین همسایگان، که استفاده از آن را در مسئله تشخیص دستخط توجیه می کند عبارتند از:

- سادگی الگوریتم
- تفسیر بسیار ساده
- دقت بالا
- چند منظوره بودن
- قابلیت استفاده در طیف وسیعی از مسایل

این الگوریتم معایبی هم دارد که عبارتند از:

- زمان متوسط
- محاسبه کمی گران است

- نیازمند حافظه زیاد چون باید تمامی داده های قبلی را ذخیره کند
- حساس به مقیاس داده
- اگر K عدد بزرگی شود پیش بینی کند و زمان افزایش پیدا می کند

در نهایت با توجه به جمیع جوانب این الگوریتم هم برای تشخیص دستخط در این پروژه انتخاب شده است.

بخش چهارم: نحوه پیاده سازی

زبان و پارادایم برنامه نویسی

پروژه به زبان پایتون Python نوشته شده و برای پیاده سازی مدل مخفی مارکوف و الگوریتم نزدیک ترین همسایگان به ترتیب از کتابخانه های استاندارد hmmlearn نسخه 0.2.5 و sklearn نسخه 0.24.2 استفاده شده است.

پیاده سازی مدل مخفی مارکوف

همانگونه که اشاره شده در این پروژه برای برای حل مسئله کلاس بندی اعداد دست نوشته با مدل مخفی مارکوف از کتابخانه hmmlearn استفاده شده است.

پس از بارگذاری مجموعه داده، لازم است از آن جا که تصاویر به صورت دو بعدی (۲۸*۲۸ پیکسل) درج شده اند، تبدیل به یک بعدی انجام گیرد. در ساده ترین حالت، کلیه پیکسل های تصاویر سطر به سطر به هم متصل شده و تشکیل یک بعد با ۷۸۴ ویژگی (feature) که به عنوان توالی مشاهده ها (observations) از آن ها استفاده خواهیم کرد، می دهد. در حالت های پیشرفته تر، تصویر به قسمت هایی تقسیم می شود که chunk نام دارد و در مدل های مخفی مارکوفی که قرار است ساخته شود هر قسمت معادل یک وضعیت (state) در نظر گرفته خواهد شد.

پس از آن به منظور شناسایی هر یک از اعداد ۰ تا ۹ یک مدل مخفی مارکوف مختص آن عدد تولید شده است که قرار است با نمونه هایی از همان عدد آموزش داده شود. از کلاس GaussianHMM برای تولید مدل ها استفاده می شود، که بر اساس آزمون و خطا نتایج مناسب تری (نسبت به MultinomialHMM و GMMHMM) ارائه می کند. پارامتر تعداد وضعیت (n_components) برای ساده ترین حالت به همان حالت پیش فرض ۱ درنظر گرفته شده است. همانگونه که اشاره شد برای حالتی که تصویر به قسمت های متعدد تقسیم می شود تعداد وضعیت باید تعداد chunk ها در نظر گرفته شود.

هر یک از مدل های تولید شده به منظور استخراج پارامتر بوسیله تابع fit با کلیه نمونه های آموزشی که برچسبی مساوی عدد آن مدل دارند آموزش داده می شوند.

برای کلیه نمونه های آزمایشی، احتمال وقوع مشاهده مربوطه با هر یک از ۱۰ مدل مذکور بوسیله تابع score محاسبه می شود و مدلی که بیشترین احتمال را تولید می کند، به عنوان برچسب آن نمونه انتخاب خواهد شد.

پیاده سازی الگوریتم نزدیک ترین همسایگان

آماده سازی مجموعه داده برای این الگوریتم هم مشابه مدل مخفی مارکوف است. برای پیاده سازی الگوریتم از کلاس KNeighborsClassifier از کتابخانه sklearn استفاده شده است. کلیه داده های آموزشی به همراه برچسب های آن ها با پارامتر n_neighbors=5 (پیش فرض) توسط تابع fit به مدل آموزش داده می شود و پس از آن از طریق تابع predict داده های آزمایشی مورد تست قرار می گیرند.

ساختار پروژه

هر یک از الگوریتم های مورد بحث در فایل جداگانه ای تحت عناوین hmm.py و knn.py پیاده سازی شده است.

بخش پنجم: روش ارزیابی

برای ارزیابی نتیجه تست ها در این پروژه، به دلیل چند کلاسه بودن مسئله، فقط از معیار دقت (Precision) استفاده شده است که از تقسیم True Positive (پیش بینی صحیح برچسب نمونه های آزمایشی) به تعداد کل نمونه های آزمایشی به دست می آید:

$$Precision = \frac{TP}{TP + FP}$$

در هر یک از الگوریتم های پروژه، برای هر یک از اعداد ۰ تا ۹ معیار دقت مدل ها جداگانه محاسبه شده و یک دقت کلی نیز از مجموع پیش بینی های نادرست مدل به همه داده های آزمایشی بدست آمده است.

بخش ششم: ارائه نتایج

دقت مدل مخفی مارکوف

بر اساس آزمایش، دقت کلی مدل مخفی مارکوف در ساده ترین حالت (1 chunk) برابر ۵۴,۶۶ درصد ارزیابی شده است که برای هر یک از اعداد ۰ تا ۹ این دقت به صورت زیر است:

عدد	دقت پیش بینی
0	۸۸,۰۶
1	۹۴,۹۸
2	۲۴,۱۳
3	۳۲,۹۷
4	۱۴,۸۷
5	۴,۴۸
6	۹۳,۵۳
7	۲۵,۷۸
8	۶۵,۵۰
9	۹۴,۹۵

دقت الگوریتم نزدیک ترین همسایگان

بر اساس آزمایش، دقت کلی الگوریتم نزدیک ترین همسایگان ۹۷,۰۵ درصد ارزیابی می شود که علت آن تناسب بالا با ماهیت صورت مسئله می باشد. از مزایای استفاده از الگوریتم نزدیک ترین همسایگان می توان به عدم نیاز به پیش فرض در خصوص مجموعه داده ها و از معایب آن به سرعت پایین پیش بینی و نیز هزینه بالای حافظه ذخیره سازی کلیه نمونه های آموزشی اشاره

کرد.

برای هر یک از اعداد ۰ تا ۹ این دقت به صورت زیر است:

عدد	دقت پیش بینی
0	۹۹,۳۹
1	۹۹,۸۲
2	۹۶,۵۱
3	۹۶,۶۳
4	۹۶,۷۴
5	۹۶,۳۰
6	۹۸,۵۴
7	۹۶,۴۰
8	۹۳,۸۴
9	۹۵,۹۴