تشخیص کارت‌های هویتی و مالی مانند تشخیص کارت ملی، عابر بانک، چک صیادی، و کپچای عددی

فهرست

فصل اول: کلیات

1. روش‌های متنوع هوش مصنوعی برای انواع تشخیص
2. پردازش تصاویر هویتی و مالی، مانند تشخیص کارت ملی، چک صیادی، تشخیص عکس‌های پرسنلی، تحلیل کارت‌های عابر بانک، و حتی حل تست‌های CAPTCHA

فصل دوم: ادبیات کار

1. مقدمه:اهمیت تشخیص کارت‌های هویتی و مالی
2. هدف اصلی پروژه:جلوگیری از تقلب و سیستماتیک کردن
3. اهمیت امنیت و حفاظت از داده‌ها در سیستم مالی و بانکی
4. چگونگی استفاده از هوش مصنوعی برای تشخیص
5. دلایل سود اوری و اهمیت بانکی
6. چالش ها در ایران

فصل سوم: روش کار

1. شبکه عصبی هوش مصنوعی
2. الگوریتم ها
3. تکنولوژی های مورد استفاده
4. پایتون
5. OpenCV
6. Pytorch
7. کد QR
8. Base64
9. مدل‌های از قبل آموزش‌دیده: ResNet18
10. شبکه عصبی کانولوشنال (CNN)
11. پیش پردازش
12. بارگذاری داده ها
13. ارزیابی یک مدل یادگیری ماشین
14. مشخص کردن تابع ضرر، بهینه ساز و متریک ارزیابی
15. آموزش مدل

فصل چهارم: کاری که انجام شده

1. تست و برسی نرم افزار
2. خروجی نرم افزار
3. نمونه در دنیا
4. موارد استفاده در حال حاضر به صورت عملی و واقعی
5. جای بهبود و توسعه
6. نرم افزار های مشابه
7. سخت افزار های مورد نیاز

فصل پنجم: نتیجه گیری

1. نتیجه گیری
2. منابع

در این سمینار، به بررسی روش‌های متنوع هوش مصنوعی می‌پردازیم که در پردازش تصاویر هویتی و مالی، مانند تشخیص کارت ملی، چک صیادی، تشخیص عکس‌های پرسنلی، تحلیل کارت‌های عابر بانک، و حتی حل تست‌های CAPTCHA عددی در وب‌سایت‌ها مورد استفاده قرار می‌گیرند. این موضوعات به صورت تئوری و علمی بررسی می‌شوند و همچنین نحوه نوشتن و اجرای کدها برای این روش‌های هوش مصنوعی نیز مورد بررسی و آموزش قرار می‌گیرد.

داده‌های مورد استفاده برای هر پروژه به صورت اختصاصی و بسته به نوع پروژه متغیر است. به عنوان مثال، در پروژه‌های ملی و کپچا، داده‌های بزرگی جمع‌آوری شده‌اند، در حالی که پروژه‌های دیگر به تعداد کمی داده نیاز داشته‌اند.

برای توسعه این پروژه‌ها، از ابزارها و زبان‌های برنامه‌نویسی مختلف استفاده شده است. به عنوان مثال، نرم‌افزار پایچارم و زبان برنامه‌نویسی پایتون به همراه کتابخانه‌های مفیدی مانند پایتورچ و cv2 مورد استفاده قرار گرفته‌اند.

با استفاده از این ابزارها، توانایی تشخیص نوع عکس ورودی که به کدام دسته تعلق دارد به صورت یک API توسعه داده شده است.

شبکه عصب ی مصنوعی با استفاده از ویژگ یهای مربوط به تصویر جهت تشخیص پیاده سازی شد.

مقدمه

اهمیت تشخیص کارت‌های هویتی و مالی

تشخیص کارت‌های هویتی و مالی از اهمیت بسیار بالایی در جوامع مدرن برخوردار است. این اهمیت به دلیل تأثیرات گسترده‌ای که تقلب و سوءاستفاده در این زمینه‌ها می‌توانند داشته باشند، بیش از پیش مورد توجه قرار گرفته است. تشخیص دقیق و سریع کارت‌های هویتی و مالی نه تنها به حفظ امنیت افراد و سازمان‌ها کمک می‌کند، بلکه اقتصاد جهانی را نیز تحت تأثیر قرار می‌دهد.

از آنجا که فناوری‌های تشخیص کارت‌های هویتی و مالی به سرعت پیشرفت کرده و روش‌های تقلب نیز پیچیده‌تر شده‌اند، نیاز به تحقیقات و مقالات علمی در این زمینه افزایش یافته است. این مقالات می‌توانند به توسعه روش‌های بهتری برای تشخیص کارت‌های هویتی و مالی کمک کرده و برای جامعه علمی و صنعتی ارزش افزوده فراهم آورند.

بنابراین، این مقدمه به معرفی موضوع اصلی مقاله پرداخته و اهمیت تشخیص کارت‌های هویتی و مالی در دنیای امروز را مورد تاکید قرار می‌دهد.

توضیح اهداف پروژه و اهمیت استفاده از شبکه‌های عصبی.

در مقاله‌ی اینجانب با موضوع تشخیص کارت ملی، عابر بانک، چک صیادی، و کپچای عددی با استفاده از هوش مصنوعی و شبکه‌های عصبی، اهداف و اهمیت استفاده از این فناوری‌ها را بیان می‌نمایم:

هدف اصلی پروژه: اصلی‌ترین هدف این پروژه افزایش امنیت و دقت در تشخیص و تصدیق هویت افراد و اسناد است. این شامل تشخیص کارت ملی افراد، اعتبارسنجی عابران بانک، تشخیص چک‌های صحیح از صیادی، و کاهش تأثیر کپچاهای عددی بر روی کاربران وبسایت‌ها می‌شود.

اهمیت استفاده از هوش مصنوعی: استفاده از هوش مصنوعی در این پروژه امکان اتوماسیون و افزایش سرعت تشخیص و اعتبارسنجی را فراهم می‌کند. همچنین، امکان تشخیص الگوهای تقلبی و تخلفات را بهبود می‌بخشد.

چگونه هوش مصنوعی می تواند به عنوان یک راه حل موثر برای تشخیص کارت های اعتباری و هویتی مورد استفاده قرار گیرد:  
هوش مصنوعی می تواند به عنوان یک راه حل موثر برای تشخیص کارت های اعتباری و هویتی مورد استفاده قرار گیرد. با استفاده از الگوریتم های هوش مصنوعی، می توان به طور خودکار کارت های اعتباری و هویتی را تشخیص داد. برای مثال، با استفاده از شبکه های عصبی، می توان به طور خودکار اطلاعات موجود در کارت های اعتباری و هویتی را تشخیص داد. همچنین، با استفاده از الگوریتم های یادگیری ماشین، می توان به طور خودکار اطلاعات موجود در کارت های اعتباری و هویتی را تشخیص داد. با استفاده از هوش مصنوعی، می توان به طور خودکار از کلاهبرداری های اینترنتی جلوگیری کرد و امنیت در صنعت پرداخت را افزایش داد. همچنین، با استفاده از هوش مصنوعی، می توان هزینه های مربوط به تشخیص کارت های اعتباری و هویتی را کاهش داد.

اهمیت شبکه‌های عصبی: برای تولید مدل برای تشخیص کارت‌های اعتباری و هویتی، شبکه‌های عصبی می‌توانند به عنوان یکی از روش‌های موثر مورد استفاده قرار گیرند. [شبکه‌های عصبی با استفاده از الگوریتم‌های یادگیری عمیق، به طور خودکار اطلاعات موجود در کارت‌های اعتباری و هویتی را تشخیص داده و به عنوان یک راه حل موثر برای کاهش جرایم مالی و کلاهبرداری‌های اینترنتی مورد استفاده قرار می‌گیرند](https://journals.iau.ir/article_685420.html" \t "_blank)[1](https://blog.faradars.org/%D8%B4%D8%A8%DA%A9%D9%87-%D8%B9%D8%B5%D8%A8%DB%8C-%DA%86%DB%8C%D8%B3%D8%AA/)[2](https://journals.iau.ir/article_685420.html). [شبکه‌های عصبی می‌توانند با استفاده از الگوریتم‌های یادگیری عمیق، به طور خودکار اطلاعات موجود در کارت‌های اعتباری و هویتی را تشخیص داده و به عنوان یک راه حل موثر برای کاهش هزینه‌های مربوط به تشخیص کارت‌های اعتباری و هویتی مورد استفاده قرار می‌گیرند](https://journals.iau.ir/article_685420.html" \t "_blank)[1](https://blog.faradars.org/%D8%B4%D8%A8%DA%A9%D9%87-%D8%B9%D8%B5%D8%A8%DB%8C-%DA%86%DB%8C%D8%B3%D8%AA/)[2](https://journals.iau.ir/article_685420.html). برای تولید مدل‌های شبکه‌های عصبی، می‌توان از الگوریتم‌های یادگیری عمیق نظیر شبکه‌های کانولوشنی (Convolutional Neural Networks) و شبکه‌های بازگشتی (Recurrent Neural Networks) استفاده کرد. [شبکه‌های کانولوشنی برای پردازش تصاویر و شبکه‌های بازگشتی برای پردازش داده‌های دنباله‌ای مانند متن و صدا به کار می‌روند](https://amanjacademy.com/neural-networks-in-deep-learning-comprehensive-guide/" \t "_blank)[3](https://amanjacademy.com/neural-networks-in-deep-learning-comprehensive-guide/).

اهداف فرعی: علاوه بر تشخیص و اعتبارسنجی، اهداف فرعی شامل کاهش تقلب، افزایش امنیت اطلاعات شخصی، و افزایش راحتی کاربران در تعامل با سیستم‌ها و وبسایت‌ها می‌باشد.

اهمیت امنیت و حفاظت از داده‌ها: با توجه به اطلاعات حساس مرتبط با کارت ملی، حساب‌های بانکی، و داده‌های شخصی، امنیت و حفاظت از داده‌ها از اهمیت بالایی برخوردار است.

تأثیر مثبت در جامعه و اقتصاد: پیاده‌سازی موفق این پروژه می‌تواند تأثیرات مثبتی بر روی جامعه و اقتصاد داشته باشد، از جمله کاهش جرایم مالی و تقلب.

با توجه به اهداف و اهمیت مذکور، این پروژه بهبود قابل توجهی در امنیت و اعتبارسنجی اسناد و اطلاعات حیاتی ارائه می‌دهد.

تکنولوژی های مورد استفاده

پایتون یک زبان برنامه نویسی سطح بالا است که در سال های اخیر محبوبیت زیادی به دست آورده است. به طور گسترده در حوزه های مختلف مانند توسعه وب، علم داده، هوش مصنوعی و غیره استفاده می شود. محبوبیت پایتون را می توان به سادگی، خوانایی و سهولت استفاده نسبت داد. این یک زبان منبع باز است، به این معنی که استفاده و توزیع آن رایگان است. ماهیت منبع باز پایتون منجر به ایجاد جامعه وسیعی از توسعه دهندگان شده است که در توسعه و نگهداری آن سهیم هستند. محبوبیت پایتون همچنین به ایجاد فرصت های شغلی متعدد برای توسعه دهندگان منجر شده است. طبق نظرسنجی انجام شده توسط Stack Overflow، پایتون یکی از محبوب ترین زبان های برنامه نویسی در بین توسعه دهندگان است که بیش از 41 درصد از توسعه دهندگان از آن استفاده می کنند. پایتون یک انتخاب عالی برای کسانی است که به دنبال یادگیری یک زبان برنامه نویسی جدید یا بهبود مهارت های خود هستند.

OpenCV یک کتابخانه نرم افزار بینایی کامپیوتر و یادگیری ماشینی رایگان و منبع باز با بیش از 2500 الگوریتم بهینه شده است. می توان از آن برای تشخیص و تشخیص چهره ها، شناسایی اشیاء، طبقه بندی اعمال انسان در فیلم ها، ردیابی حرکات دوربین، ردیابی اجسام متحرک، استخراج مدل های سه بعدی اشیاء، تولید ابرهای نقطه سه بعدی از دوربین های استریو، دوختن تصاویر به یکدیگر برای تولید تصویری با وضوح بالا استفاده کرد. از کل یک صحنه، تصاویر مشابه را از یک پایگاه داده تصویر پیدا کنید، چشم های قرمز را از تصاویر گرفته شده با فلاش حذف کنید، حرکات چشم را دنبال کنید، مناظر را تشخیص دهید و نشانگرهایی برای پوشاندن آن با واقعیت افزوده ایجاد کنید، و غیره. این به طور گسترده در صنعت برای واقعی استفاده می شود زمان برنامه های بینایی کامپیوتر به زبان C++ نوشته شده است و رابط اصلی آن به زبان C++ است، اما همچنان برای پایتون، جاوا، متلب و سایر زبان‌ها پیوندهایی دارد. در رمان خود، می توانید از OpenCV برای توصیف ادراک بصری یک شخصیت یا ایجاد یک نقطه طرح استفاده کنید. به عنوان مثال، می توانید توضیح دهید که چگونه یک شخصیت از OpenCV برای تشخیص حضور یک فرد در اتاق استفاده می کند یا چگونه از OpenCV برای ردیابی حرکت ماشین استفاده می شود. همچنین می‌توانید از OpenCV برای ایجاد یک نقطه طرح استفاده کنید، با استفاده از یک شخصیت از OpenCV برای حل یک مشکل یا غلبه بر یک مانع.

PyTorch یک کتابخانه محبوب یادگیری ماشین منبع باز برای یادگیری عمیق است که به طور گسترده توسط محققان و توسعه دهندگان استفاده می شود. به دلیل سهولت استفاده، انعطاف پذیری و مقیاس پذیری شناخته شده است. PyTorch در بالای پایتون ساخته شده است که ادغام با سایر کتابخانه ها و چارچوب های پایتون را آسان می کند. PyTorch طیف گسترده ای از ابزارها و ماژول ها را ارائه می دهد که می توانند برای ساخت و آموزش مدل های پیچیده یادگیری ماشین استفاده شوند. این ابزارها شامل عملیات تانسور، تمایز خودکار و ماژول های شبکه عصبی است. PyTorch همچنین تعدادی مدل از پیش آموزش دیده را ارائه می دهد که می توانند برای کارهای مختلفی مانند طبقه بندی تصویر، تشخیص اشیا و پردازش زبان طبیعی استفاده شوند.

PyTorch یک جامعه بزرگ و فعال از توسعه دهندگان دارد که به کتابخانه کمک می کنند و برای سایر کاربران پشتیبانی می کنند. همچنین منابع زیادی به صورت آنلاین در دسترس هستند، مانند آموزش، مستندات، و انجمن ها، که می توانند به کاربران کمک کنند تا PyTorch را شروع کنند و نحوه استفاده موثر از آن را بیاموزند.

به طور خلاصه، PyTorch یک کتابخانه یادگیری ماشینی قدرتمند و انعطاف پذیر است که می تواند برای ساخت و آموزش مدل های پیچیده استفاده شود. سهولت استفاده و ادغام آن با سایر کتابخانه های پایتون، آن را به یک انتخاب محبوب در میان محققان و توسعه دهندگان تبدیل کرده است.

کد QR یا کد پاسخ سریع، یک بارکد دو بعدی است که توسط دوربین گوشی هوشمند قابل خواندن است. این توسط Denso Wave، یکی از شرکت‌های تابعه ژاپنی تامین‌کننده Toyota Denso، برای تسریع فرآیندهای لجستیکی برای تولید خودروهای خود توسعه داده شد. کدهای QR اکنون به طور گسترده در بازاریابی تلفن همراه با پذیرش گسترده تلفن های هوشمند استفاده می شوند.

کدهای QR می توانند اطلاعات بسیار بیشتری نسبت به بارکدهای سنتی ذخیره کنند. آنها می توانند تا 7089 کاراکتر عددی، 4296 کاراکتر الفبایی عددی یا 2953 بایت داده باینری را ذخیره کنند. کدهای QR را می توان برای ذخیره URL ها، اطلاعات تماس، متن، پیامک، ایمیل و حتی بیت کوین 12 استفاده کرد.

برای ایجاد یک کد QR، می توانید از ابزار تولید کننده کد QR استفاده کنید. این ابزارها به شما این امکان را می دهند که کدهای QR را برای نیازهای خود مانند URL، مخاطب، ایمیل، تلفن، پیامک یا کارت مجازی ایجاد و سفارشی کنید. همچنین می‌توانید کدهای QR را با وب‌کم خود اسکن کنید و محتوای موجود در آن‌ها را دریافت کنید

Base64 یک طرح رمزگذاری باینری به متن است که برای نمایش داده های باینری در قالب رشته ای اسکی استفاده می شود. این برای انتقال داده های ذخیره شده در قالب های باینری در کانال هایی طراحی شده است که فقط محتوای متنی را به طور قابل اعتماد پشتیبانی می کنند. Base64 به ویژه در شبکه جهانی وب رایج است، جایی که یکی از کاربردهای آن امکان جاسازی فایل های تصویری یا سایر دارایی های باینری در دارایی های متنی مانند فایل های HTML و CSS است. Base64 همچنین به طور گسترده برای ارسال پیوست های ایمیل استفاده می شود. فرآیند رمزگذاری شامل گرفتن یک ورودی باینری و تقسیم آن به گروه های 6 بیتی است. سپس هر گروه 6 بیتی به یک کاراکتر در الفبای Base64 نگاشت می شود که از 64 کاراکتر تشکیل شده است. خروجی حاصل رشته ای از کاراکترهای ASCII است که می تواند از طریق کانال های مبتنی بر متن منتقل شود. Base64 یک طرح رمزگذاری پرکاربرد است که کاربردهای زیادی در برنامه نویسی کامپیوتر دارد. این یک روش ساده و کارآمد برای نمایش داده های باینری در قالب متنی است که می تواند از طریق کانال هایی که فقط محتوای متنی را پشتیبانی می کنند منتقل شود.

مدل‌های از قبل آموزش‌دیده:

شده از قبل روی مجموعه داده‌های بزرگ آموزش داده می‌شوند و سپس برای استفاده دیگران در دسترس قرار می‌گیرند. آنها مزایای زیادی نسبت به روش‌های آموزشی سنتی دارند، مانند سریع‌تر بودن، دقیق‌تر بودن و در دسترس بودن بیشتر.

در کتابخانه PyTorch، بسته فرعی torchvision.models شامل تعاریفی از مدل‌ها برای پرداختن به وظایف مختلف است، از جمله طبقه‌بندی تصویر، تقسیم‌بندی معنایی پیکسلی، تشخیص شی، تقسیم‌بندی نمونه، تشخیص نقطه کلید شخص، طبقه‌بندی ویدیو و جریان نوری 3.

استفاده از مدل های از پیش آموزش دیده می تواند در زمان و منابع صرفه جویی کند، دقت را بهبود بخشد و دسترسی به مدل های پیشرفته را فراهم کند. پروژه 2.

TorchVision با استفاده از PyTorch torch.hub وزن های از پیش آموزش دیده را برای هر معماری ارائه شده ارائه می دهد. نمونه‌برداری از یک مدل از پیش آموزش‌دیده، وزن‌های آن را در فهرست حافظه پنهان دانلود می‌کند. این دایرکتوری را می توان با استفاده از متغیر محیطی TORCH\_HOME 3 تنظیم کرد.

توجه به این نکته ضروری است که مدل های از پیش آموزش دیده ارائه شده در این کتابخانه ممکن است مجوزها یا شرایط و ضوابط خاص خود را داشته باشند که از مجموعه داده مورد استفاده برای آموزش مشتق شده است. این مسئولیت شماست که تعیین کنید آیا مجوز استفاده از مدل ها را برای مورد استفاده خود دارید یا خیر.

راه اندازی مدل های از پیش آموزش دیده ساده است. از نسخه 0.13، TorchVision یک API پشتیبانی چند وزنی جدید برای بارگذاری وزن های مختلف به روش های سازنده مدل موجود 3 ارائه می دهد.

شبکه عصبی کانولوشنال (CNN) نوعی شبکه عصبی است که معمولاً در برنامه‌های بینایی رایانه استفاده می‌شود. این شبکه برای پردازش داده‌هایی طراحی شده است که ساختار شبکه‌ای مانند تصاویر دارند و به ویژه در شناسایی الگوها و ویژگی‌ها مؤثر است. در داده های بصری 1.

یک CNN از یک یا چند لایه کانولوشن تشکیل شده است که مجموعه ای از فیلترها را برای استخراج ویژگی های 1 بر روی داده های ورودی اعمال می کند. سپس خروجی هر لایه کانولوشن از یک لایه ادغام عبور می کند که ابعاد داده ها را کاهش می دهد و به جلوگیری از آن کمک می کند. overfitting 1. در نهایت خروجی آخرین لایه ادغام از یک یا چند لایه کاملاً متصل عبور داده می شود که وظیفه طبقه بندی یا رگرسیون نهایی 1 را انجام می دهد.

برای ایجاد یک شبکه عصبی کانولوشن، باید تعداد لایه‌های کانولوشن، تعداد فیلترها در هر لایه، اندازه فیلترها، نوع لایه جمع‌آوری و تعداد لایه‌های کاملا متصل را انتخاب کنید. باید در مورد تابع فعال سازی برای استفاده، تابع ضرر برای بهینه سازی و الگوریتم بهینه سازی برای استفاده تصمیم گیری شود.

در اینجا مراحل کلی برای ایجاد یک شبکه عصبی کانولوشن وجود دارد:

بارگذاری داده ها: داده های آموزشی و آزمایشی را در حافظه بارگذاری کنید.

پیش پردازش داده ها: داده ها را با عادی سازی مقادیر پیکسل، تغییر اندازه تصاویر و تقسیم داده ها به مجموعه های آموزشی و اعتبار سنجی، از قبل پردازش کنید.

ایجاد مدل: یک مدل متوالی ایجاد کنید و لایه های کانولوشنال، ادغام و کاملا متصل را اضافه کنید.

کامپایل مدل: با تعیین تابع ضرر، بهینه ساز و معیار ارزیابی، مدل را کامپایل کنید.

آموزش مدل: آموزش مدل بر روی داده های آموزشی و اعتبارسنجی آن بر روی داده های اعتبار سنجی.

ارزیابی مدل: مدل را بر روی داده های آزمون ارزیابی کنید و دقت را محاسبه کنید.

منابع زیادی به صورت آنلاین در دسترس هستند که اطلاعات دقیق تری در مورد نحوه ایجاد یک شبکه عصبی کانولوشن ارائه می دهند، از جمله آموزش ها و نمونه های کد.

ResNet18 یک معماری شبکه عصبی کانولوشن است که توسط تحقیقات مایکروسافت در سال 2015 معرفی شد. این یک نوع از مدل های خانواده ResNet است که به دلیل توانایی خود در آموزش شبکه های عصبی بسیار عمیق بدون رنج از مشکل گرادیان ناپدید شدن 1 شناخته شده است. مدل ResNet18 از 18 لایه شامل 16 لایه کانولوشنال و 2 لایه کاملاً متصل 1 تشکیل شده است.

مدل ResNet18 بر روی مجموعه داده ImageNet از قبل آموزش داده شده است، که مجموعه بزرگی از تصاویر برچسب گذاری شده است که معمولا برای آموزش مدل های بینایی کامپیوتری استفاده می شود. سپس روی یک مجموعه داده کوچکتر برای یک کار خاص تنظیم شود.

یکی از مزایای اصلی مدل ResNet18 دقت آن است. در چالش تشخیص تصویری در مقیاس بزرگ ImageNet (ILSVRC) 2012، مدل ResNet18 به نرخ خطای بالای 5 برابر با 5.25% دست یافت که به طور قابل توجهی بهتر از آخرین مدل قبلی 1 بود.

در مقایسه با مدل‌های دیگر، مانند AlexNet و VGG، مدل ResNet18 از نظر محاسباتی گران‌تر است و به پارامترهای کمتری نیاز دارد.

از نظر تفاوت‌ها با مدل‌های دیگر، مدل ResNet18 از اتصالات باقی‌مانده برای رسیدگی به مشکل گرادیان ناپدید 1 استفاده می‌کند. این به مدل اجازه می‌دهد تا ویژگی‌های پیچیده‌تری را بیاموزد و شبکه‌های عمیق‌تری را نسبت به مدل‌های دیگر آموزش دهد.

به طور کلی، مدل ResNet18 یک معماری شبکه عصبی کانولوشنال قدرتمند و کارآمد است که به طور گسترده در برنامه های بینایی کامپیوتر استفاده می شود.

پیش پردازش یک مرحله مهم در یادگیری ماشینی است که شامل تمیز کردن و تبدیل داده های خام به قالبی است که می تواند برای آموزش مدل 1 استفاده شود. کیفیت داده های مورد استفاده برای آموزش یک مدل تأثیر قابل توجهی بر عملکرد مدل دارد و پیش پردازش به آن کمک می کند. برای اطمینان از کیفیت بالا و مناسب بودن داده ها برای کار در دست 1.

دلایل متعددی وجود دارد که چرا پیش پردازش برای آموزش یک مدل مهم است. اول، با حذف اطلاعات نامربوط یا اضافی 1، به کاهش نویز در داده ها کمک می کند. این می تواند دقت مدل را بهبود بخشد و خطر بیش از حد برازش 1 را کاهش دهد. یا توزیع 1. این می تواند یادگیری مدل را از داده ها آسان تر کند و عملکرد آن را بهبود بخشد. 1. سوم، پیش پردازش می تواند به مدیریت داده های از دست رفته با وارد کردن یا حذف مقادیر از دست رفته 1 کمک کند. این می تواند از خطاها و سوگیری های موجود در مدل جلوگیری کند. با داده های از دست رفته 1.

مراحل پیش پردازش خاص مورد نیاز برای یک کار معین به ماهیت داده ها و الزامات مدل 1 بستگی دارد. با این حال، برخی از مراحل پیش پردازش متداول شامل تمیز کردن داده ها، عادی سازی داده ها، مقیاس بندی ویژگی، انتخاب ویژگی، و کاهش ابعاد 1 است.

به طور خلاصه، پیش پردازش یک مرحله ضروری در یادگیری ماشین است که به اطمینان از اینکه داده های مورد استفاده برای آموزش یک مدل با کیفیت بالا و مناسب برای کار در دست هستند کمک می کند. با کاهش نویز، عادی سازی داده ها و مدیریت داده های از دست رفته، پیش پردازش می تواند دقت و عملکرد مدل 1 را بهبود بخشد.

پیش پردازش یک مرحله مهم در یادگیری ماشینی است که شامل تمیز کردن و تبدیل داده های خام به قالبی است که می تواند برای آموزش مدل 1 استفاده شود. کیفیت داده های مورد استفاده برای آموزش یک مدل تأثیر قابل توجهی بر عملکرد مدل دارد و پیش پردازش به آن کمک می کند. برای اطمینان از کیفیت بالا و مناسب بودن داده ها برای کار در دست 1.

دلایل متعددی وجود دارد که چرا پیش پردازش برای آموزش یک مدل مهم است. اول، با حذف اطلاعات نامربوط یا اضافی 1، به کاهش نویز در داده ها کمک می کند. این می تواند دقت مدل را بهبود بخشد و خطر بیش از حد برازش 1 را کاهش دهد. یا توزیع 1. این می تواند یادگیری مدل را از داده ها آسان تر کند و عملکرد آن را بهبود بخشد. 1. سوم، پیش پردازش می تواند به مدیریت داده های از دست رفته با وارد کردن یا حذف مقادیر از دست رفته 1 کمک کند. این می تواند از خطاها و سوگیری های موجود در مدل جلوگیری کند. با داده های از دست رفته 1.

مراحل پیش پردازش خاص مورد نیاز برای یک کار معین به ماهیت داده ها و الزامات مدل 1 بستگی دارد. با این حال، برخی از مراحل پیش پردازش متداول شامل تمیز کردن داده ها، عادی سازی داده ها، مقیاس بندی ویژگی، انتخاب ویژگی، و کاهش ابعاد 1 است.

به طور خلاصه، پیش پردازش یک مرحله ضروری در یادگیری ماشین است که به اطمینان از اینکه داده های مورد استفاده برای آموزش یک مدل با کیفیت بالا و مناسب برای کار در دست هستند کمک می کند. با کاهش نویز، عادی سازی داده ها و مدیریت داده های از دست رفته، پیش پردازش می تواند دقت و عملکرد مدل 1 را بهبود بخشد.

آموزش مدل یادگیری ماشینی شامل چندین مرحله است، از جمله بارگذاری داده ها، پیش پردازش داده ها، ایجاد مدل، کامپایل مدل، آموزش مدل و ارزیابی مدل 123.

هنگامی که داده ها بارگیری و پیش پردازش شدند، مرحله بعدی ایجاد مدل است. برای یک شبکه عصبی کانولوشن (CNN)، این معمولاً شامل تعیین تعداد لایه‌های کانولوشن، تعداد فیلترها در هر لایه، اندازه فیلترها، نوع لایه جمع‌آوری و تعداد لایه‌های کاملاً متصل 12 است.

پس از ایجاد مدل، باید با مشخص کردن تابع ضرر، بهینه ساز و متریک ارزیابی 12 کامپایل شود. مدل برای به حداقل رساندن ضرر 12. معیار ارزیابی برای اندازه گیری عملکرد مدل در طول آموزش 12 استفاده می شود.

هنگامی که مدل کامپایل شد، می توان آن را بر روی داده های آموزشی با استفاده از روش fit() 12 آموزش داد. در طول آموزش، مدل وزن های خود را برای به حداقل رساندن تابع ضرر و بهبود عملکرد خود تنظیم می کند.

پس از آموزش مدل، می توان آن را بر روی داده های آزمون با استفاده از روش ()value 12 ارزیابی کرد. این روش مقدار معیار ارزیابی را برای مدل در داده های آزمون 12 برمی گرداند.

به طور خلاصه، آموزش یک مدل یادگیری ماشینی شامل چندین مرحله است، از جمله بارگذاری و پیش پردازش داده ها، ایجاد و کامپایل مدل، آموزش مدل، و ارزیابی مدل. هر یک از این مراحل برای اطمینان از اینکه مدل قادر به یادگیری از داده ها و پیش بینی دقیق است، مهم است.

ارزیابی یک مدل یادگیری ماشینی یک گام اساسی در فرآیند توسعه مدل است. این کمک می‌کند تا مشخص شود که چقدر مدل می‌تواند به داده‌های جدید تعمیم دهد و پیش‌بینی‌های دقیق انجام دهد. 1. چندین روش مختلف برای ارزیابی یک مدل یادگیری ماشینی وجود دارد، از جمله اعتبار سنجی نگهدارنده، اعتبارسنجی متقاطع k-fold، و بوت استرپینگ 12.

در اعتبار سنجی holdout، مجموعه داده به دو بخش تقسیم می شود: یک مجموعه آموزشی و یک مجموعه تست 1. مدل بر روی مجموعه آموزشی آموزش داده می شود و سپس بر روی مجموعه آزمایشی 1 ارزیابی می شود. این روش ساده و آسان برای پیاده سازی است، اما می تواند به نحوه تقسیم داده ها 1 حساس باشید.

در اعتبار سنجی متقاطع k-fold، مجموعه داده به k زیر مجموعه با اندازه مساوی 1 تقسیم می شود. مدل بر روی زیر مجموعه های k-1 آموزش داده می شود و در زیر مجموعه 1 باقی مانده ارزیابی می شود. این فرآیند k بار تکرار می شود و هر زیر مجموعه دقیقا یک بار استفاده می شود. به عنوان داده های اعتبارسنجی 1. این روش قوی تر از اعتبار سنجی نگهدارنده است، اما می تواند از نظر محاسباتی گران باشد.

در بوت استرپ، چندین نمونه بوت استرپ با نمونه برداری تصادفی از مجموعه داده با جایگزینی 2 ایجاد می شود. مدل بر روی هر نمونه بوت استرپ آموزش داده می شود و بر روی مجموعه داده اصلی 2 ارزیابی می شود. این روش زمانی مفید است که مجموعه داده کوچک باشد یا زمانی که مدل از نظر محاسباتی گران است. قطار 2.

انتخاب روش ارزیابی به نیازهای خاص مسئله و ماهیت داده ها بستگی دارد. به عنوان دقت، دقت، یادآوری و امتیاز F1 1.

به طور خلاصه، ارزیابی یک مدل یادگیری ماشین گام مهمی در فرآیند توسعه مدل است. چندین روش مختلف برای ارزیابی یک مدل وجود دارد، از جمله اعتبار سنجی holdout، اعتبارسنجی متقاطع k-fold و bootstrapping. انتخاب روش ارزیابی به نیازهای خاص مسئله و ماهیت داده ها بستگی دارد. گزارش نتایج با استفاده از معیارهای مناسب برای اطمینان از اینکه مدل قادر به تعمیم به داده های جدید و پیش بینی های دقیق است مهم است.

حال به خود کد میپردازیم:

device = "cuda" if torch.cuda.is\_available() else "cpu"

بررسی می‌کند که آیا کتابخانه CUDA در سیستم موجود است یا نه، رشته «cuda» را به دستگاه متغیر در صورت وجود، اختصاص می‌دهد، در غیر این صورت رشته «cpu» 1 را اختصاص می‌دهد.

CUDA یک پلت فرم محاسباتی موازی و مدل برنامه نویسی است که توسط NVIDIA توسعه یافته است که به توسعه دهندگان اجازه می دهد تا از قدرت GPU ها برای کارهای محاسباتی همه منظوره استفاده کنند. 1.

با اختصاص مقدار "cuda" به متغیر دستگاه در صورت در دسترس بودن CUDA، کد تضمین می کند که مدل در صورت امکان از GPU برای محاسبات استفاده می کند، که می تواند به طور قابل توجهی روند آموزش را سرعت بخشد. 1 اگر CUDA در دسترس نباشد، کد سقوط می کند بازگشت به استفاده از CPU برای محاسبات 1.

به طور خلاصه، خط کدی که ارائه کرده‌اید بررسی می‌کند که آیا کتابخانه CUDA در سیستم موجود است یا نه، رشته «cuda» را به متغیر دستگاه اختصاص می‌دهد، در غیر این صورت رشته «cpu» را اختصاص می‌دهد. این به مدل اجازه می دهد تا در صورت امکان از GPU برای محاسبات استفاده کند، که می تواند عملکرد آموزش را بهبود بخشد.

اندازه گیری زمان صرف شده برای آموزش یک مدل یادگیری ماشینی به چند دلیل مهم است. اول، می تواند به ما در بهینه سازی عملکرد مدل با شناسایی تنگناها یا مناطقی که آموزش مدل 1 بیش از حد طول می کشد، کمک کند. دوم، می تواند به ما در مقایسه عملکرد مدل های مختلف یا پیکربندی های مختلف یک مدل 1 کمک کند در نهایت، می تواند به ما کمک کند تا زمان مورد نیاز برای آموزش مدل بر روی مجموعه داده های بزرگتر یا با معماری های پیچیده تر را تخمین بزنیم.

به طور خلاصه، استفاده از start=time.time() برای اندازه گیری زمان صرف شده برای آموزش یک مدل یادگیری ماشینی یک تکنیک مهم برای بهینه سازی عملکرد، مقایسه مدل ها و تخمین زمان آموزش است. با ثبت زمان شروع فرآیند و محاسبه زمان سپری شده در پایان، می توان مدت زمان تکمیل فرآیند را تعیین کرد و زمینه های بهبود را شناسایی کرد.

خواندن یک تصویر در حالت مقیاس خاکستری تکنیک رایجی است که در بینایی کامپیوتر و برنامه های پردازش تصویر استفاده می شود. 12. تصاویر در مقیاس خاکستری تنها از یک کانال رنگی استفاده می کنند که معمولاً تنها با استفاده از 8 بیت 1 نمایش داده می شود. نسبت به تصاویر تمام رنگی 1.

تصویری را از فایلی به نام img\_name می‌خواند و با استفاده از پرچم 3 cv2.IMREAD\_GRAYSCALE آن را به مقیاس خاکستری تبدیل می‌کند. این در هنگام کار با تصاویری که به اطلاعات رنگی نیاز ندارند، مانند هنگام انجام تشخیص لبه یا سایر ویژگی‌ها مفید است. وظایف استخراج 12.

تبدیل یک تصویر به مقیاس خاکستری همچنین می تواند با حذف اطلاعات رنگی نامربوط یا اضافی 1 به کاهش میزان نویز در تصویر کمک کند. این کار می تواند دقت مدل را بهبود بخشد و خطر بیش از حد برازش 1 را کاهش دهد.

به طور خلاصه، خواندن یک تصویر در حالت خاکستری یک تکنیک رایج است که در بینایی کامپیوتر و برنامه های پردازش تصویر استفاده می شود. تبدیل تصویر به مقیاس خاکستری می تواند به کاهش نویز و بهبود دقت مدل 12 کمک کند.

تبدیل تصویر یک تکنیک مهم در بینایی کامپیوتر است که شامل اصلاح تصویر ورودی برای بهبود عملکرد یک مدل یادگیری ماشینی 1 است. انواع مختلفی از تبدیل تصویر وجود دارد که بسته به نیازهای خاص مشکل و ماهیت می توان از آنها استفاده کرد. از داده ها 12.

برخی از انواع متداول تبدیل تصویر عبارتند از:

تغییر اندازه: تغییر اندازه یک تصویر شامل تغییر ابعاد آن به یک اندازه خاص است. این می تواند برای استاندارد کردن اندازه تصاویر در یک مجموعه داده یا برای کاهش هزینه محاسباتی پردازش تصاویر بزرگ مفید باشد.

مقیاس خاکستری: مقیاس خاکستری یک تصویر شامل تبدیل آن به مقیاس خاکستری است که فقط از یک کانال رنگ استفاده می کند. این می تواند برای کاهش میزان نویز در تصویر و تسهیل کار با 1 مفید باشد.

عادی سازی: عادی سازی یک تصویر شامل مقیاس بندی مقادیر پیکسل آن در محدوده خاصی است. این می تواند برای بهبود عملکرد یک مدل یادگیری ماشینی با اطمینان از اینکه داده های ورودی دارای محدوده ثابتی از مقادیر 1 هستند مفید باشد.

افزایش داده‌ها: تقویت داده‌ها شامل تولید نمونه‌های آموزشی جدید با اعمال تبدیل‌های تصادفی به داده‌های ورودی، مانند چرخش، چرخش یا برش تصویر است. این می تواند برای افزایش اندازه مجموعه داده آموزشی و بهبود عملکرد تعمیم مدل 1 مفید باشد.

استخراج ویژگی: استخراج ویژگی شامل استخراج ویژگی های مفید از تصویر ورودی است که می تواند برای آموزش یک مدل یادگیری ماشین استفاده شود. این را می توان با استفاده از تکنیک هایی مانند تشخیص لبه، تشخیص گوشه یا تشخیص لکه 3 انجام داد.

به طور خلاصه، تبدیل تصویر یک تکنیک مهم در بینایی کامپیوتر است که شامل اصلاح تصویر ورودی برای بهبود عملکرد یک مدل یادگیری ماشینی است. بسته به نیازهای خاص مشکل و ماهیت داده، انواع مختلفی از تبدیل تصویر وجود دارد که می توان از آنها استفاده کرد. برخی از انواع متداول تبدیل تصویر عبارتند از: تغییر اندازه، مقیاس خاکستری، عادی سازی، افزایش داده ها و استخراج ویژگی ها

کلاس DataLoader در PyTorch ابزاری است که یک قابل تکرار را روی یک مجموعه داده 1 ارائه می دهد. از آن برای بارگذاری داده ها به صورت دسته ای در طول فرآیند آموزش استفاده می شود که می تواند به بهبود عملکرد مدل و کاهش نیازهای حافظه کمک کند.

خط کدی که ارائه کردید یک شی DataLoader ایجاد می‌کند که شیء مجموعه داده را می‌پیچد و اندازه دسته‌ای 32 را مشخص می‌کند و داده‌ها را در حین آموزش 2 به هم می‌ریزد. این به این معنی است که مدل بر روی دسته‌هایی از 32 نمونه در یک زمان آموزش داده می‌شود. نمونه ها در طول هر دوره 2 تصادفی می شوند.

استفاده از DataLoader می تواند به بهبود عملکرد مدل با کاهش مقدار حافظه مورد نیاز برای بارگذاری داده ها و با فعال کردن استفاده از تکنیک های پردازش موازی 1 کمک کند. همچنین می تواند با اطمینان از اینکه آموزش به بهبود دقت مدل کمک کند. داده ها به ترتیب تصادفی در طول هر دوره 1 به مدل ارائه می شود.

به طور خلاصه، کلاس DataLoader در PyTorch ابزاری است که یک داده قابل تکرار را روی یک مجموعه داده ارائه می دهد و برای بارگذاری داده ها به صورت دسته ای در طول فرآیند آموزش استفاده می شود. استفاده از DataLoader می تواند به بهبود عملکرد و دقت مدل با کاهش نیازهای حافظه و امکان استفاده از تکنیک های پردازش موازی کمک کند.

برای تنظیم دقیق مدل، می‌توانید برخی از لایه‌ها را منجمد کنید و برخی دیگر را روی یک مجموعه داده جدید ۱ آموزش دهید. برای مثال، می‌توانید همه لایه‌ها را به جز چند لایه آخر ثابت کنید و فقط آن لایه‌ها را روی مجموعه داده جدید ۱ آموزش دهید. این می تواند به سرعت بخشیدن به روند تمرین و کاهش خطر اضافه کردن 1 کمک کند.

به طور خلاصه، بارگذاری یک مدل از پیش آموزش‌دیده و تنظیم دقیق آن با فریز کردن برخی از لایه‌ها و آموزش برخی دیگر، تکنیک رایجی است که در یادگیری انتقال استفاده می‌شود. کدی که ارائه کردید یک مدل ResNet18 از پیش آموزش دیده را بارگیری می کند و آخرین لایه کاملاً متصل را با یک لایه جدید که دارای یک ویژگی خروجی واحد است جایگزین می کند. برای تنظیم دقیق مدل، می‌توانید برخی از لایه‌ها را منجمد کنید و برخی دیگر را روی یک مجموعه داده جدید آموزش دهید.

تابع ضرر یک تابع ریاضی است که تفاوت بین خروجی پیش‌بینی‌شده یک مدل یادگیری ماشینی و خروجی واقعی ۱ را اندازه‌گیری می‌کند. هدف تابع ضرر، به حداقل رساندن این تفاوت است که به عنوان ضرر یا هزینه ۱ نیز شناخته می‌شود. انتخاب تابع ضرر به نیازهای خاص مسئله و ماهیت داده بستگی دارد.

در وظایف طبقه‌بندی باینری، از دست دادن آنتروپی متقاطع باینری و BCEWithLogitsLoss معمولاً از توابع ضرر استفاده می‌شوند. از دست دادن و از دست دادن softmax معمولاً از توابع ضرر استفاده می شوند.

اهمیت تابع ضرر در توانایی آن برای اندازه گیری عملکرد مدل یادگیری ماشین و هدایت فرآیند بهینه سازی است. داده 1.

به طور خلاصه، تابع ضرر یک تابع ریاضی است که تفاوت بین خروجی پیش‌بینی‌شده یک مدل یادگیری ماشینی و خروجی واقعی را اندازه‌گیری می‌کند. انتخاب تابع ضرر به نیازهای خاص مشکل و ماهیت داده بستگی دارد. اهمیت تابع ضرر در توانایی آن برای اندازه‌گیری عملکرد مدل یادگیری ماشین و هدایت فرآیند بهینه‌سازی است.

بهینه ساز الگوریتمی است که برای به روز رسانی وزن ها و بایاس های یک مدل یادگیری ماشین در طول آموزش به منظور به حداقل رساندن تابع ضرر 1 استفاده می شود. انتخاب بهینه ساز به نیازهای خاص مسئله و ماهیت داده ها بستگی دارد.

برخی از انواع رایج بهینه سازهای مورد استفاده در یادگیری عمیق عبارتند از:

نزول گرادیان تصادفی (SGD): این یک بهینه ساز ساده است که وزن ها و بایاس های مدل را بر اساس گرادیان تابع تلفات با توجه به پارامترهای 1 به روز می کند. از نظر محاسباتی کارآمد است و پیاده سازی آن آسان است، اما می تواند آهسته باشد. همگرا می شوند و ممکن است در حداقل محلی 1 گیر کنند.

Momentum: این بهینه ساز از میانگین متحرک گرادیان ها برای به روز رسانی وزن ها و بایاس های مدل 1 استفاده می کند. این به تسریع همگرایی مدل کمک می کند و می تواند به غلبه بر حداقل های محلی 1 کمک کند.

Adagrad: این بهینه ساز نرخ یادگیری مدل را بر اساس فراوانی ویژگی ها در داده های ورودی 1 تطبیق می دهد. برای داده های پراکنده مفید است و می تواند به بهبود عملکرد مدل در ویژگی های کمیاب 1 کمک کند.

RMSprop: این بهینه‌ساز از میانگین متحرک گرادیان‌های مجذور برای تطبیق نرخ یادگیری مدل 1 استفاده می‌کند. برای مشکلات غیر ثابت مفید است و می‌تواند به بهبود عملکرد مدل در داده‌های نویز 1 کمک کند.

Adam: این بهینه‌ساز ایده‌های سرعت و نرخ یادگیری تطبیقی را برای به‌روزرسانی وزن‌ها و سوگیری‌های مدل 1 ترکیب می‌کند. از نظر محاسباتی کارآمد است و نشان داده شده است که روی طیف گسترده‌ای از مسائل به خوبی کار می‌کند.

انتخاب بهینه ساز به عوامل مختلفی بستگی دارد، از جمله اندازه مجموعه داده، پیچیدگی مدل، و ماهیت داده ها. حرکت، که سهم گرادیان های قبلی را در به روز رسانی 1 کنترل می کند.

به طور خلاصه، بهینه‌ساز الگوریتمی است که برای به‌روزرسانی وزن‌ها و سوگیری‌های یک مدل یادگیری ماشینی در طول آموزش به منظور به حداقل رساندن تابع ضرر استفاده می‌شود. انواع مختلفی از بهینه سازها در یادگیری عمیق مورد استفاده قرار می گیرند که هر کدام نقاط قوت و ضعف خاص خود را دارند. انتخاب بهینه ساز به عوامل مختلفی از جمله اندازه مجموعه داده، پیچیدگی مدل و ماهیت داده بستگی دارد. برخی از پارامترهای مهم بهینه سازها شامل نرخ یادگیری و تکانه 1 است.

برای حلقه زدن لایه‌های یک مدل در PyTorch، می‌توانید از متد model.named\_parameters() استفاده کنید که یک تکرارکننده بر روی پارامترهای نام‌گذاری‌شده مدل برمی‌گرداند. سپس می‌توانید روی لایه‌های مدل تکرار کنید و با استفاده از ویژگی name پارامتر به پارامترهای هر لایه دسترسی پیدا کنید.

یک مدل یادگیری ماشین را با استفاده از بهینه ساز Adam و آنتروپی متقاطع باینری با از دست دادن لجیت آموزش می دهد. روش optimizer.zero\_grad() برای تنظیم گرادیان تمام پارامترهای مدل قبل از محاسبه گرادیان برای دسته فعلی استفاده می شود. متد loss.backward() برای محاسبه گرادیان های تابع ضرر با توجه به پارامترهای مدل استفاده می شود. در نهایت، روش optimizer.step() برای به روز رسانی پارامترهای مدل بر اساس گرادیان های محاسبه شده استفاده می شود.

به طور خلاصه، برای حلقه زدن لایه های یک مدل در PyTorch، می توانید از متد model.named\_parameters() استفاده کنید. متدهای optimizer.zero\_grad()، loss.backward() و optimizer.step() برای محاسبه و به روز رسانی گرادیان پارامترهای مدل در طول آموزش استفاده می شوند.

ذخیره یک مدل PyTorch در دیسک و بارگذاری آن بعداً یک تکنیک رایج است که در یادگیری ماشین استفاده می شود. تابع ()torch.save برای ذخیره وضعیت یک مدل PyTorch در یک فایل روی دیسک 1 استفاده می شود. از متد model.state\_dict() برای به دست آوردن وضعیت فعلی مدل استفاده می شود که شامل مقادیر همه موارد قابل یادگیری است. مولفه های .

تابع ()torch.load برای بارگذاری وضعیت یک مدل PyTorch از یک فایل روی دیسک استفاده می شود. متد model.load\_state\_dict() برای بارگذاری وضعیت مدل در شی مدل استفاده می شود. پارامتر map\_location برای تعیین دستگاهی که مدل باید در آن بارگذاری شود استفاده می شود.

از متد model.state\_dict() برای به دست آوردن وضعیت فعلی مدل استفاده می شود، که سپس با استفاده از تابع ()torch.save در فایلی به نام model12.pth ذخیره می شود. بعداً از تابع ()torch.load برای بارگذاری وضعیت مدل از فایلی به نام model11.pth استفاده می‌شود که سپس با استفاده از متد model.load\_state\_dict() در شی مدل بارگذاری می‌شود.

به طور خلاصه، ذخیره یک مدل PyTorch در دیسک و بارگذاری آن بعداً یک تکنیک رایج در یادگیری ماشینی است. تابع torch.save () برای ذخیره وضعیت یک مدل PyTorch در یک فایل روی دیسک استفاده می شود، در حالی که تابع ()torch.load برای بارگذاری وضعیت یک مدل PyTorch از یک فایل روی دیسک استفاده می شود. متدهای model.state\_dict() و model.load\_state\_dict() به ترتیب برای بدست آوردن و تنظیم وضعیت مدل استفاده می شوند.

در نهایت، تابع predict\_image با مسیر تصویری که باید طبقه بندی شود فراخوانی می شود.

این کد یک فایل به صورت base66 دریافت میتواند کند و در نهایت با تشخیص نوع مدرک بارگذاری شده میتواند به امنیت و شناسایی مدارک به صورت کامپیوتری کمک کند.