مجله محاسبات نرم، جلد 13، شماره 2 (پاییز و زمستان 1403)، صفحه 42ـ55
https://doi.org/10.22052/scj.2024.243236.1021

مجله محاسبات نرم
SOFT COMPUTING JOURNAL
تارنمای مجله: scj.kashanu.ac.ir

دانشگاه کاشان
University of Kashan

تشخیص ارقام دست‌نویس انگلیسی با استفاده از شبکه عصبی کانولوشن✧

مهسا بهرامیان[1]، دانشجوی کارشناسی ارشد، آرش عظیم‌زاده ایرانی[1]، استادیار، رضا پورقلی[1*]، استاد، احمد علیاری بروجنی[1]، دکتری
[1] دانشکده ریاضی و علوم کامپیوتر، دانشگاه دامغان، دامغان، ایران.

اطلاعات مقاله

تاریخچه مقاله:
دریافت 3 آبان ماه 1400
پذیرش 1 آبان ماه 1402

کلمات کلیدی:
تشخیص ارقام دست‌نویس
شبکه عصبی کانولوشن
مجموعه داده MNIST
بینایی ماشین
پردازش تصویر

چکیده

انسان‌ها می‌توانند با استفاده از چشم‌ها و مغز خودشان جهان اطراف خود را ببینند و حس کنند. بینایی رایانه بر روی توانایی کامپیوترها برای دیدن و پردازش تصاویر به همان روشی که انسان‌ها استفاده می‌کنند، کار می‌کند. هدف از کار ما ایجاد مدلی است که بتواند ارقام دست‌نویس انگلیسی را از تصویر اصلی با دقت زیاد شناسایی کند. هدف ما این است که با استفاده از مفاهیم شبکه عصبی کانولوشن (Convolutional Neural Network) و مجموعه داده MNIST، این کار را انجام دهیم. در این کار، هدف ما یادگیری و بکارگیری عملی مفاهیم شبکه‌های عصبی کانولوشن است. اگرچه هدف ما ایجاد مدلی است که بتواند ارقام دست‌نویس را تشخیص دهد، اما می‌توانیم آن را برای حروف و دست خط یک شخص دیگر نیز گسترش دهیم. در این کار توانستیم در آزمایش‌های خود به بهبود قابل قبولی نسبت به کارهای انجام شده قبلی برسیم که به دست آوردن دقت 99/30% نشان‌دهنده موفقیت روش پیشنهادی برای تشخیص ارقام دست‌نویس مجموعه داده MNIST می‌باشد.

1. مقدمه

یادگیری عمیق یکی از زیررشته‌های یادگیری ماشین است که در آن سعی می‌شود انتزاعات و ویژگی‌های سطح بالای موجود در داده‌ها با استفاده از معماری‌های سلسله مراتبی فرا گرفته شود. شبکه عصبی کانولوشن[1] عمیق نیز یکی از مهم‌ترین و پرکاربردترین الگوریتم‌های مورد استفاده در یادگیری عمیق است. این روش، یک روش نوظهور بوده و به صورت گسترده‌ای در دامنه‌های مختلفی از یادگیری ماشین و به خصوص بینایی کامپیوتر مورد استفاده قرار گرفته است. اخیرا شبکه‌های عصبی کانولوشن (CNN) به یکی از جذاب‌ترین رویکردها تبدیل شده‌اند و عامل اصلی موفقیت‌های اخیر در مباحث یادگیری ماشین مانند تقسیم‌بندی تصویر و تشخیص چهره هستند. لذا، در کار حاضر شبکه عصبی کانولوشن برای کار طبقه‌بندی تصویر انتخاب شده است. ما می‌توانیم از این شبکه برای شناسایی ارقام دست‌نویس که یکی از موارد مهم معاملات علمی و تجاری است، استفاده کنیم. در اهداف واقعی زندگی ما کاربردهای زیادی برای شناسایی رقم دست‌نویس وجود دارد، ما می‌توانیم از آن در بانک‌ها برای خواندن چک‌ها، در دفاتر پستی برای

✧ نوع مقاله: پژوهشی
* نویسنده مسئول
پست(های) الکترونیک: bahramiannmahsa@gmail.com (بهرامیان)
a.azimzadeh@du.ac.ir (عظیم‌زاده ایرانی)
pourgholi@du.ac.ir (پورقلی)
aliyari@du.ac.ir (علیاری بروجنی)

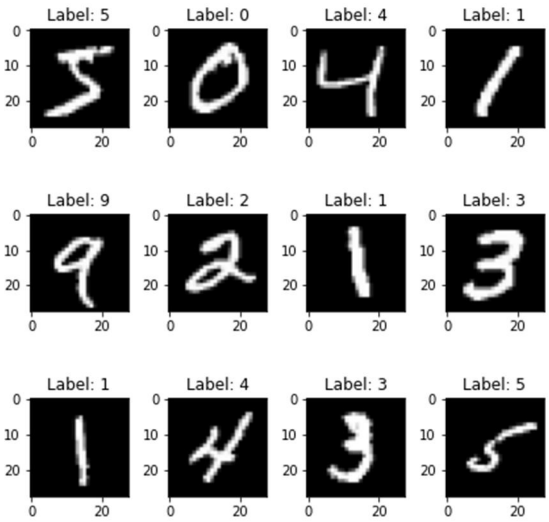
[1] Convolution

نحوه ارجاع به مقاله: مهسا بهرامیان، آرش عظیم‌زاده ایرانی، رضا پورقلی، احمد علیاری بروجنی، «تشخیص ارقام دست‌نویس انگلیسی با استفاده از شبکه عصبی کانولوشن»، مجله محاسبات نرم، جلد 13، شماره 2، ص42ـ55، پاییز و زمستان 1403.

مرتب‌سازی نامه‌ها و بسیاری از کارهای دیگر استفاده کنیم.

1.1. مجموعه داده MNIST

MNIST یک مجموعه داده از ارقام دست‌نویس انگلیسی است. از این مجموعه داده می‌توان برای آموزش سیستم‌های مختلف پردازش تصویر استفاده کرد. همچنین از این مجموعه داده برای آموزش و آزمایش در زمینه یادگیری ماشین بسیار استفاده می‌شود. این مجموعه داده دارای 60000 مثال برای آموزش و 10000 مثال برای آزمایش است. تصاویر موجود در این مجموعه داده نمونه دست خط اسکن شده 250 نفر است که نیمی از آنها کارمندان اداره سرشماری ایالات متحده و نیمی از آنها دانش‌آموزان دبیرستان بودند. هر تصویر موجود در این مجموعه داده دارای اندازه ثابت 28×28 پیکسل است. این مجموعه داده برای افرادی مناسب است که می‌خواهند ضمن صرف حداقل تلاش برای پیش‌پردازش و قالب‌بندی، تکنیک‌های یادگیری و روش‌های تشخیص الگو را بر روی داده‌های دنیای واقعی امتحان کنند. در این کار از این مجموعه داده استفاده شده است که نمونه‌ای از آن در شکل (1) نشان داده شده است.

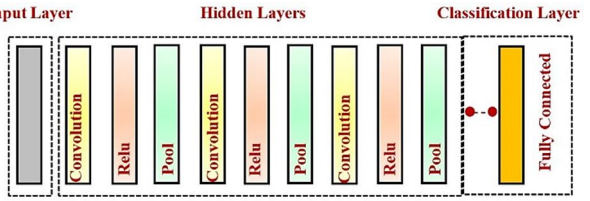


شکل (1): تعدادی از تصاویر مجموعه داده MNIST

2.1. شبکه عصبی کانولوشن

شبکه‌های عصبی کانولوشن، شبکه‌های عصبی مصنوعی عمیق هستند و یکی از مهم‌ترین روش‌های یادگیری عمیق محسوب می‌شوند. این شبکه‌ها نوعی از شبکه‌های عصبی مصنوعی پیشخور [1] می‌باشند. استفاده از روش شبکه‌های عصبی کانولوشن بسیار کارآمد بوده و یکی از رایج‌ترین روش‌ها در کاربردهای مختلف بینایی کامپیوتر است. وقتی کسی یادگیری عمیق را با یک شبکه عصبی شروع می‌کند، متوجه خواهد شد که شبکه عصبی کانولوشن ابزاری مهم برای طبقه‌بندی تصاویر است. از شبکه عصبی کانولوشن می‌توان برای طبقه‌بندی تصاویر، خوشه‌بندی عکس‌ها بر اساس شباهت، شناسایی شی در یک فضا، شناسایی چهره‌ها، افراد، علائم خیابان‌ها، تومورها، پلاک‌ها و بسیاری از موارد دیگر استفاده کرد.

ساختار اصلی CNN شامل یک لایه ورودی، چند لایه مخفی و یک لایه خروجی است. در شکل (2) یک مدل معماری شبکه عصبی کانولوشن نشان داده شده است که شامل لایه ورودی، چندین لایه پنهان (تکرار لایه‌های کانولوشن، نرمال‌سازی، پولینگ[2]) و یک لایه کاملا متصل و یک لایه خروجی است [1].



شکل (2): یک مدل معماری شبکه عصبی کانولوشن [1]

شرح مفصل لایه‌های موجود در یک شبکه عصبی کانولوشن در زیر آمده است.

1.2.1. لایه ورودی

داده‌های ورودی بارگیری و در لایه ورودی ذخیره می‌شوند. این لایه ارتفاع، عرض و تعداد کانال‌های (اطلاعات RGB) تصویر ورودی را توصیف می‌کند [1].

2.2.1. لایه پنهان

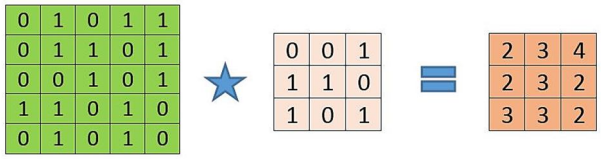
لایه‌های پنهان در شبکه عصبی کانولوشن فرآیند استخراج ویژگی

[1] Feed-forward
[2] Pooling

را انجام می‌دهند که در آن از مجموعه‌ای از عملیات کانولوشن، پولینگ و توابع فعال‌سازی استفاده می‌شود. ویژگی‌های متمایز ارقام دست‌نویس در این مرحله شناسایی می‌شود [1].

3.2.1. لایه کانولوشن

لایه کانولوشن، قسمت اصلی ساخت یک شبکه عصبی کانولوشن است. لایه کانولوشن اولین لایه قرار گرفته در بالای تصویر ورودی است و برای استخراج ویژگی‌های یک تصویر استفاده می‌شود [1]. در شکل (3) نمونه‌ای از عملیات کانولوشن نشان داده شده است [2]. در این شکل اندازه تصویر ورودی یک ماتریس 5 × 5، اندازه فیلتر 3 × 3 و اندازه خروجی برابر با 3 × 3 = (5–3+1) × (5–3+1) است.



شکل (3): عملیات کانولوشن [2]

4.2.1. لایه پولینگ

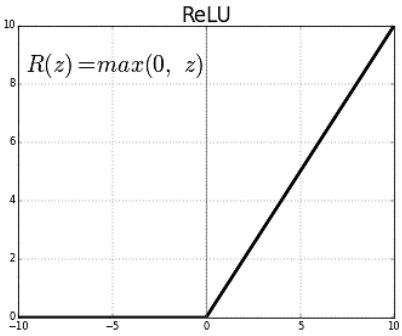
هنگام ساخت شبکه عصبی کانولوشن، برای کاهش ابعاد ورودی و بنابراین کاهش پیچیدگی محاسباتی، یک لایه پولینگ بین دو لایه کانولوشن اضافه می‌شود [1]. عملیات max-pooling یکی از رایج‌ترین روش‌های پولینگ می‌باشد. فرآیند max-pooling در شکل (4) نشان داده شده است [1].

شکل (4): عملیات MAX-POOLING با اندازه فیلتر 2×2 و گام 2 [1]

5.2.1. لایه فعال‌سازی

همانند معماری منظم شبکه‌های عصبی، معماری شبکه‌های عصبی کانولوشن نیز شامل توابع فعال‌سازی[1] برای معرفی غیرخطی بودن در سیستم است [1]. توابع ReLu[2] (واحد خطی اصلاح شده) و Softmax (بیشینه هموار) برخی از گزینه‌های معروف در میان توابع فعال‌سازی مختلف هستند که به طور گسترده در مدل‌های یادگیری عمیق مورد استفاده قرار می‌گیرند [1]. در ادامه به بیان جزئیات آنها می‌پردازیم.

تابع فعال‌سازی ReLu: یکی از توابع فعال‌سازی مورد استفاده در کار حاضر، تابع غیرخطی واحد خطی اصلاح شده (ReLu) است که برای مقادیر ورودی کمتر از صفر (منفی) خروجی صفر و برای مقادیر ورودی بیشتر از صفر (مثبت) خروجی برابر با ورودی می‌باشد (شکل (5) را مشاهده کنید) [1].



شکل (5): تابع فعال‌سازی ReLU

تابع فعال‌سازی Softmax: از تابع فعال‌سازی بیشینه هموار (Softmax) غالبا به عنوان آخرین تابع فعال‌سازی در یک شبکه عصبی برای نرمال‌سازی خروجی شبکه و تبدیل آن به توزیع احتمال استفاده می‌شود. نرمال‌سازی در این حالت نسبت به کلاس‌های خروجی پیش‌بینی شده، صورت می‌گیرد. در تمامی ساختارهای کانولوشنی آخرین لایه تماما متصل به یک لایه Softmax متصل می‌شود. در حقیقت لایه Softmax کار طبقه‌بندی در شبکه‌های کانولوشنی را انجام می‌دهد. این لایه شامل تعدادی نورون برابر با تعداد کلاس‌های مساله دسته‌بندی است.

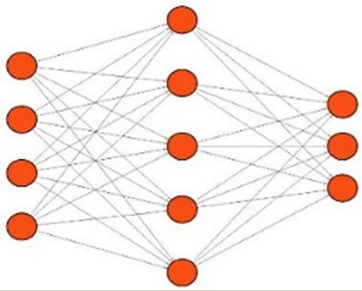
6.2.1. لایه طبقه‌بندی

لایه طبقه‌بندی آخرین لایه در معماری CNN است که یک شبکه از نوع تغذیه رو به جلو و کاملا متصل است که به طور

[1] Activation Function

[2] Rectified linear unit

عمده به عنوان طبقه‌بندی‌کننده پذیرفته می‌شود. سلول‌های عصبی در لایه‌های کاملا متصل به تمام سلول‌های عصبی لایه قبلی متصل هستند (شکل (6)). این لایه کلاس‌های پیش‌بینی شده را با شناسایی تصویر ورودی محاسبه می‌کند که این کار با ترکیب تمام ویژگی‌های یاد گرفته شده توسط لایه‌های قبلی انجام می‌شود. تعداد کلاس‌های خروجی به تعداد کلاس‌های موجود در مجموعه داده هدف بستگی دارد [1].



شکل (6): لایه‌های کاملا متصل [2]

در کار حاضر، لایه طبقه‌بندی از تابع فعال‌سازی Softmax برای طبقه‌بندی ویژگی‌های تولید شده از تصویر ورودی دریافت شده از لایه قبلی در کلاس‌های مختلف بر اساس داده‌های آموزشی استفاده می‌کند.

2. کارهای مرتبط

در حوزه کار حاضر، شبکه عصبی عمیق (DNN[1]) یک برنامه شناخته شده برای تشخیص تصویر، طبقه‌بندی اشیا، تشخیص گفتار و سایر موارد است و این قابلیت را دارد که دقت طبقه‌بندی بالاتری را ارائه دهد. شبکه عصبی کانولوشن نیز بخشی از رویکرد یادگیری عمیق است. کارهای مهم زیادی در شبکه‌های عصبی کانولوشن برای شناسایی ارقام دست‌نویس انجام شده است [3]- [6]. زمینه‌های تحقیقاتی زیادی همانند شناخت برخط، شناخت غیربرخط، تشخیص دست خط در زمان واقعی، تایید امضا، تفسیر آدرس پستی، پردازش چک‌های بانکی و تشخیص نویسنده وجود دارد.

نویسندگان مقاله [7] یک مدل مبتنی بر CNN را برای تشخیص ارقام دست‌نویس مجموعه داده MNIST پیشنهاد کردند که مدل آنها شامل هشت لایه بود. آنها ادعا کردند که معماری پیشنهادی آنها می‌تواند دقت 98/85% را در 8569 ثانیه ارائه دهد. مدل CNN دیگری برای مجموعه داده MNIST توسط نویسندگان در مقاله [8] ارائه شد که مدل پیشنهادی شامل هفت لایه بود. این مدل شامل یک لایه ورودی و یک لایه خروجی و پنج لایه مخفی در وسط بود. بهترین دقت با تغییر تعداد لایه‌های پنهان مدل و تعداد دوره‌ها ارزیابی شد و در نهایت بهترین دقت به دست آمده 99/21% با تعداد دوره 15 بود. سه رویکرد متفاوت DNN، DBN[2]، CNN توسط نویسندگان در مقاله [9] برای تشخیص ارقام دست‌نویس پیشنهاد شد. قبل از آموزش شبکه عصبی، پیش‌پردازش، تقسیم‌بندی و استخراج ویژگی‌ها انجام شد. آنها بهترین دقت 98/08% را با استفاده از روش DNN پیدا کردند. مدل CNN دیگری با هفت لایه توسط نویسندگان در مقاله [10] برای مجموعه داده MNIST پیشنهاد شد. آنها دقت آزمایش 95/7% را در 500 دوره از مدل CNN پیشنهادی خود پیدا کردند. معماری ترکیبی CNN به همراه Deeplearning4j (DL4J) برای شناسایی ارقام دست‌نویس مجموعه داده MNIST توسط نویسندگان در مقاله [11] ارائه شد. آنها از هرگونه مراحل پیش‌پردازش اجتناب کردند و دقت 99/21% را از مدل خود به دست آوردند. یک سیستم تشخیص دیگر با استفاده از پرسپترون چند لایه (MLP) برای شناسایی ارقام دست‌نویس مجموعه داده MNIST توسط نویسندگان در مقاله [12] پیشنهاد شد. آنها برای اهداف آموزشی از الگوریتم پس‌انتشار[3] و برای اهداف اعتبارسنجی از شبکه‌های پیشخور استفاده کردند. آنها بهترین نتیجه را با تغییر تعداد تکرارها ارزیابی کردند. آنها از 5000 داده استفاده کردند و دقت 99/32% را در 250 تکرار به دست آوردند. برای تشخیص خودکار ارقام دست‌نویس مجموعه داده MNIST، نویسندگان در مقاله [13] یک مدل CNN با چهار لایه را پیشنهاد کردند. این مدل ساده با تعداد دوره‌های مختلف آموزش داده شد و در نهایت دقت 98% به دست آمد.

[1] Deep Neural Network

[2] Deep belief network

[3] Back propagation

طبقه‌بندی موثر دیگر ماشین بردار پشتیبان (SVM) است که برای شناسایی ارقام مجموعه داده MNIST استفاده شد [14]، [15]. Eva Tuba و Milan Tuba و Dana Simian گروهی از نویسندگان هستند که در مقاله [14] از روش SVM برای تشخیص ارقام دست‌نویس استفاده کردند و از الگوریتم خفاش [1] برای بهینه‌سازی کار خود بهره بردند که دقت نهایی کار آنها 95/60% می‌باشد. ترکیبی از سیستم SVM و CNN برای تشخیص ارقام دست‌نویس مجموعه داده MNIST در مقاله [15] پیشنهاد شد که ویژگی‌های تصاویر ارقام با استفاده از CNN استخراج شده است و از SVM نیز برای طبقه‌بندی تصاویر در لایه خروجی استفاده شده است. روش پیشنهادی آنها یک مرحله پیش‌پردازش قبل از استفاده از CNN را اعمال کرده و به دقت 99/28% رسید. X.-X. Niu و C. Y. Suen گروهی هستند که یک روش ترکیبی برای تشخیص ارقام دست‌نویس در مقاله [18] ارائه کردند. این محققان از شبکه عصبی کانولوشن (CNN) برای استخراج ویژگی‌های تصویر استفاده کردند و برای طبقه‌بندی ارقام دست‌نویس از طبقه‌بندی ترکیبی شامل CNN و SVM استفاده کردند و دقت طبقه‌بندی 94/40% را به دست آورند. در کار انجام شده توسط Chayaporn Kaensar در مقاله [19] دقت نهایی طبقه‌بندی با استفاده از روش SVM 96/93% می‌باشد. Reza Ebrahimzadeh و Mahdi Jampour گروه دیگری از نویسندگان هستند که در مقاله [20] از SVM و هیستوگرام گرادیان جهت‌دار (HOG) برای تشخیص ارقام دست‌نویس استفاده کردند و در کار خود به دقت 97/25% دست یافتند. Bouchra EL QACIMY و Mounir AIT KERROUM و Ahmed HAMMOUCH در مقاله [21] از DCT برای استخراج ویژگی‌های تصویر و از SVM برای طبقه‌بندی ارقام دست‌نویس استفاده کردند و دقت طبقه‌بندی 98/88% را به دست آورند.

با مطالعه تحقیقات یادگیری عمیق انجام شده می‌توان نتیجه گرفت که یادگیری عمیق ارائه‌دهنده عملکرد عالی برای طبقه‌بندی تصویر و همچنین طبقه‌بندی دست خط است.

[1] Bat algorithm

3. روش پیشنهادی

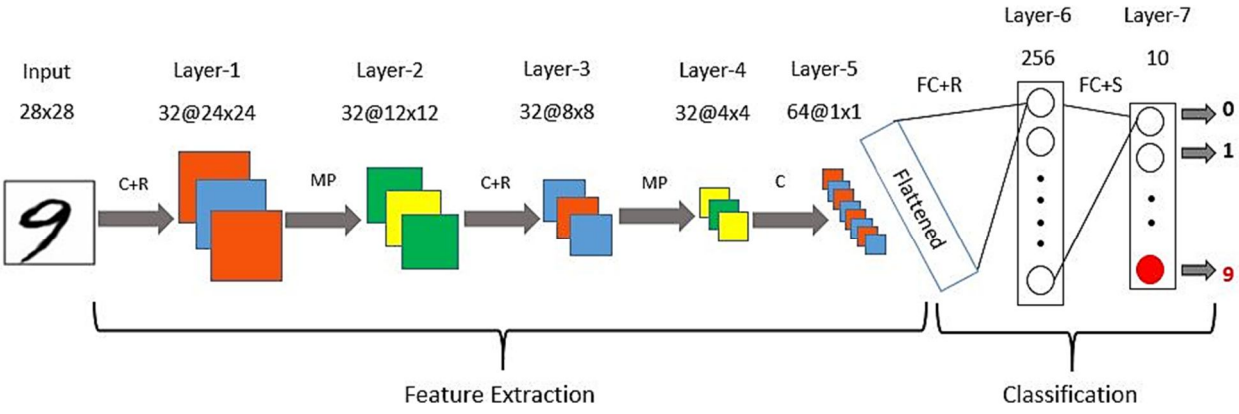
هدف کار حاضر اجرای مفهوم شبکه عصبی کانولوشن برای شناسایی ارقام دست‌نویس است. درک شبکه عصبی کانولوشن و بکارگیری آن در سیستم شناسایی ارقام دست نویس هدف کار حاضر است. شبکه‌های عصبی کانولوشن نقشه‌های ویژگی را از تصاویر دوبعدی استخراج می‌کنند، سپس می‌توانند تصاویر را با استفاده از نقشه‌های ویژگی طبقه‌بندی کنند.

شبکه عصبی کانولوشن به جای داشتن یک لایه کاملا متصل به سلول‌های عصبی، نگاشت پیکسل‌های تصویر را با فضای همسایگی در نظر می‌گیرد. شبکه عصبی کانولوشن ابزاری قدرتمند در پردازش سیگنال و تصویر است. حتی در زمینه‌های بینایی رایانه‌ای مانند تشخیص دست خط، طبقه‌بندی اشیا طبیعی و تقسیم‌بندی، شبکه عصبی کانولوشن در مقایسه با سایر ابزارها، ابزاری بسیار بهتر بوده است.

1.3. معماری مدل پیشنهادی

اکنون وقت آن رسیده است که نگاهی کلی به شبکه عصبی کانولوشن پیشنهادی خود داشته باشیم. مدل پیشنهادی کار حاضر شباهت‌هایی با سایر معماری‌های شناسایی رقم دست‌نویس دارد [3]- [6]، [16]، اما تعداد فیلترها، نورون‌ها و توابع فعال‌سازی برای عملکرد بهتر تغییر کرده است.

یک شبکه عصبی کانولوشن ساده دنباله‌ای از لایه‌ها است و هر لایه از طریق یک تابع قابل تغییر (فیلتر)، یک حجم فعال‌سازی را به حجم دیگر تبدیل می‌کند. در کار حاضر برای ساخت شبکه از سه نوع لایه اصلی استفاده شده است: لایه‌های کانولوشن، لایه‌های پولینگ و لایه‌های کاملا متصل. این لایه‌ها روی هم قرار گرفته می‌شوند تا ساختار شبکه تشکیل شود. شبکه عصبی کانولوشن پیشنهادی کار حاضر هفت لایه دارد که در شکل (7) نشان داده شده است. در ابتدا، نیاز به برخی از پیش‌پردازش‌ها بر روی تصاویر مانند تغییر اندازه تصاویر، نرمال‌سازی مقادیر پیکسل‌ها و غیره می‌باشد. پس از پیش‌پردازش لازم، داده‌ها آماده تغذیه به مدل می‌شوند.



C=Covolution, R=ReLu, MP= Max Pooling, FC=Fully Connected, S=Softmax

شکل (7): معماری مدل پیشنهادی شبکه عصبی کانولوشن ما

لایه 1 از یک لایه کانولوشن با تابع فعال‌سازی ReLu تشکیل شده است. این لایه اولین لایه کانولوشن معماری شبکه عصبی کانولوشن پیشنهادی است. این لایه تصویر پیش‌پردازش شده با اندازه 28×28 را به عنوان ورودی می‌گیرد. اندازه فیلتر کانولوشن 5×5 است.لایه‌گذاری (p)[1] صفر است (در اطراف همه طرف‌های تصویر)، گام(ها) 1 و تعداد فیلترها 32 است. بعد از عملیات کانولوشن، نقشه ویژگی با اندازه 32@24×24 ایجاد می‌شود که در آن 32 تعداد نقشه‌های ویژگی است که برابر با تعداد فیلترهای استفاده شده می‌باشد و 24 از رابطه (1) به دست می‌آید:

$$\left(\left(n+2p-f\right)/s\right)+1 \qquad (1)$$

که در آن، n اندازه تصویر ورودی، p اندازه لایه‌گذاری اعمال شده روی تصویر، f اندازه فیلتر و s اندازه گام است. سپس تابع فعال‌سازی ReLu در هر نقشه ویژگی اعمال می‌شود.

لایه 2 یک لایه Max Pooling است. این لایه ورودی با اندازه 32@24×24 را از لایه قبلی دریافت می‌کند. اندازه پولینگ 2×2 است، لایه‌گذاری صفر و گام 2 است. بعد از این عملیات Max Pooling، نقشه‌های ویژگی‌ای به اندازه 32@12×12 به دست می‌آید. Max Pooling در هر نقشه ویژگی به طور مستقل انجام می‌شود، بنابراین، ما همان تعداد نقشه‌های ویژگی لایه قبلی را داریم و مقدار 12 از همان رابطه (1) به دست می‌آید. این لایه تابع فعال‌سازی ندارد.

لایه 3 دومین لایه کانولوشن با تابع فعال‌سازی ReLu است. این لایه ورودی اندازه 32@12×12 را از لایه قبلی دریافت می‌کند. اندازه فیلتر 5×5 است. لایه‌گذاری صفر، گام 1 و تعداد فیلترها 32 است. بعد از این عملیات کانولوشن، نقشه ویژگی به اندازه 32@8×8 به دست می‌آید. سپس تابع فعال سازی ReLu در هر نقشه ویژگی اعمال می‌شود.

لایه 4 دومین لایه Max Pooling است. این لایه ورودی اندازه 32@8×8 را از لایه قبلی دریافت می‌کند. اندازه پولینگ 2×2 است، لایه‌گذاری صفر و گام 2 است. بعد از این عملیات Max Pooling، نقشه‌های ویژگی به اندازه 32@4×4 به دست می‌آید.

لایه 5 سومین لایه کانولوشن است که تابع فعال‌سازی ReLu را ندارد. این لایه ورودی اندازه 32@4×4 را از لایه قبلی دریافت می‌کند. اندازه فیلتر 4×4 است. لایه‌گذاری صفر، گام 1 و تعداد فیلترها 64 است. بعد از این عملیات کانولوشن، نقشه‌های ویژگی با اندازه 64@1×1 به دست می‌آید. این لایه به عنوان یک لایه کاملا متصل عمل می‌کند و با عملیات مسطح کردن یک بردار یک بعدی به اندازه 64 تولید می‌کند.

لایه 6 یک لایه کاملا متصل است. این لایه یک بردار یک بعدی به اندازه 64 را می‌گیرد و یک بردار یک بعدی با اندازه 256 را تولید می‌کند. این لایه همچنین تابع فعال‌سازی ReLu را دارد.

[1] Padding

لایه 7 آخرین لایه شبکه است. همچنین این لایه، یک لایه کاملا متصل است. این لایه مقدار هر کلاس را محاسبه می‌کند، در نتیجه یک بردار به اندازه 10، به طوری که هر یک از ده عدد مربوط به مقدار یک کلاس است (همانند ده دسته مجموعه داده MNIST) را داریم. برای خروجی‌های نهایی تابع فعال‌سازی softmax اعمال می‌شود.

در این روش پیشنهادی، شبکه عصبی کانولوشن تصویر اصلی را لایه به لایه از مقادیر پیکسلی اصلی به مقدار کلاس نهایی تبدیل می‌کند. باید توجه داشت که برخی از لایه‌ها حاوی پارامترهایی هستند و برخی از لایه‌های دیگر این پارامترها را ندارند. به طور خاص، لایه‌های کانولوشن و کاملا متصل، تغییراتی را انجام می‌دهند که تابعی از فعال‌سازی‌ها در حجم ورودی و پارامترهایی مثل وزن‌ها و بایاس نورون‌ها هستند. از طرف دیگر، لایه‌های پولینگ و ReLu یک تابع ثابت (پارامتر قابل تنظیم ندارند) را پیاده‌سازی می‌کنند. پارامترهای موجود در لایه‌های کانولوشن و کاملا متصل با الگوریتم نزول شیب تصادفی[2] آموزش داده می‌شوند و اصلاح می‌شوند. به طوری که مقادیر کلاس ایجاد شده برای هر تصویر با برچسب‌های مجموعه آموزش برای آن تصویر مطابقت داشته باشد. الگوریتم مدل آموزش دیده‌ای را که برای طبقه‌بندی ارقام موجود در داده‌های آزمون استفاده می‌شود، آماده می‌کند. بنابراین، می‌توان ارقام ارائه شده در تصاویر را به صورت زیر طبقه‌بندی کرد: کلاس 0 و 1 و 2 و 3 و 4 و 5 و 6 و 7 و 8 و 9.

4. پیاده‌سازی

در کار حاضر برای پیاده‌سازی معماری شبکه عصبی کانولوشن پیشنهادی، از کتابخانه‌های Keras و TensorFlow استفاده شده است. در واقع برای پیاده‌سازی شبکه‌های عصبی کانولوشن در زبان برنامه‌نویسی پایتون عناصر موجود در کتابخانه‌های Keras و TensorFlow مورد استفاده قرار می‌گیرند. برای پیاده‌سازی مدل پیشنهادی از مدل ترتیبی[3] استفاده شده است که یکی از مدل‌های موجود در کتابخانه Keras می‌باشد و به عنوان طبقه‌بندی‌کننده استفاده می‌شود و شامل دسته‌ای خطی از لایه‌ها است. کل کار ما شامل بخش‌های زیر می‌باشد:

1- آماده‌سازی داده‌ها
2- ساخت و کامپایل مدل پیشنهادی
3- آموزش و اعتبارسنجی مدل پیشنهادی
4- ارزیابی و پیش‌بینی مدل پیشنهادی
5- ذخیره مدل در حافظه برای استفاده مجدد

آماده‌سازی داده‌ها اولین قدم کار حاضر است. قبل از اینکه شبکه را بسازیم، باید داده‌های آموزش و آزمایش خود را تنظیم کنیم، داده‌ها را ترکیب کنیم، برچسب‌ها را ترکیب کرده و به اندازه مناسب تغییر شکل دهیم. ما مجموعه داده‌های نرمال شده، برچسب‌ها و اطلاعات متفرقه را ذخیره می‌کنیم.

برای ساختن و آموزش شبکه‌های عصبی کانولوشن باید پارامترهای مهم مقداردهی اولیه را تعریف کنیم. اندازه دسته (batch size)، دوره (epoch) و نرخ یادگیری (learning rate) سه پارامتر مهم در هنگام ساخت و آموزش یک مدل CNN می‌باشند که باید مقداردهی شوند. اندازه دسته تعداد نمونه‌ها را برای مرحله آموزش CNN تعیین می‌کند. CNN کلیه داده‌های آموزشی به تعداد اندازه دسته مشخص شده را پردازش می‌کند. ما برای کارایی محاسبات از اندازه دسته استفاده می‌کنیم و مقدار آن به سخت‌افزار موجود کاربر بستگی خواهد داشت. دوره یک عبور موفقیت‌آمیز به طرف جلو و یک عقبگرد از طریق شبکه است. به طور معمول مقدار آن عدد زیادی است. در صورت رضایت از همگرایی در یک حالت خاص (دوره انتخاب شده) در شبکه، می‌توانیم یک بار مقدار را کاهش دهیم. یافتن بهترین مقدار دوره یک فرآیند تجربی خواهد بود. میزان (نرخ) یادگیری یک پارامتر بسیار حساس است که مدل را به سمت همگرایی سوق می‌دهد و به وسیله آن می‌توان مقدار تغییر وزن‌ها را در شبکه تنظیم کرد. در کار حاضر، برای حداکثر دقت از اندازه دسته 40، تعداد دوره 15 و نرخ یادگیری 0/01 استفاده شده است.

[2] Stochastic gradient descent
[3] Sequential

مرحله دوم ساخت و تدوین مدل است. اکنون می‌توانیم شبکه عصبی کانولوشن خود را با ایجاد هر لایه به صورت جداگانه همان‌طور که در زیر توضیح داده شده است، ایجاد کنیم.

در ابتدا برای ایجاد مدل پیشنهادی، نوع مدل که از نوع Sequential می‌باشد را مشخص می‌کنیم.

```
# Build the CNN model
model=Sequential ()
```

در مرحله بعد نیاز است لایه‌های مدل پیشنهادی را به ترتیب اضافه کنیم که برای اضافه کردن هر لایه از model.add استفاده می‌کنیم. برای ایجاد یک لایه از نوع کانولوشن از Conv2D استفاده می‌کنیم که نیاز است به ترتیب تعداد فیلترهای مورد استفاده در لایه موردنظر، اندازه فیلتر کانولوشن (kernel_size)، اندازه گام (strides) حرکت فیلتر روی تصویر، مقدار لایه‌گذاری در اطراف تصویر، نوع تابع فعال‌سازی مورد استفاده (activation) و اندازه تصویر ورودی (input_shape) را برای ایجاد این لایه مشخص کنیم. برای ایجاد یک لایه از نوع Max Pooling از MaxPool2D استفاده می‌کنیم که نیاز است به ترتیب اندازه پولینگ (pool_size)، اندازه گام (strides) حرکت فیلتر روی تصویر و مقدار لایه‌گذاری در اطراف تصویر را برای ایجاد این لایه مشخص کنیم. در ادامه مفاهیم گام و لایه‌گذاری، که در ایجاد لایه‌ها مورد استفاده قرار می‌گیرند را توضیح خواهیم داد.

گام (stride): پارامتری است که به عنوان اندازه گام تعریف می‌شود و فیلتر هر بار با توجه به اندازه گام تعیین شده حرکت می‌کند. مثلا مقدار گام 1، حرکت فیلتر را به صورت پیکسل به پیکسل نشان می‌دهد.

لایه‌گذاری (padding): مفهوم padding برای دستیابی به دقت بیشتر در معماری CNN معرفی شده است و برای کنترل مقدار کوچک شدن خروجی لایه مورد استفاده قرار می‌گیرد. خروجی لایه‌ها یک نقشه ویژگی است که کوچک‌تر از تصویر ورودی است که این نقشه ویژگی خروجی شامل اطلاعات بیشتری در مورد پیکسل‌های میانی است و از این رو اطلاعات موجود در گوشه‌ها را از دست می‌دهد. در واقع در صورت نیاز ردیف‌ها و ستون‌های شامل مقدار صفر به دور تا دور تصویر اضافه می‌شوند تا از کوچک شدن نقشه ویژگی خروجی جلوگیری کنند. یکی از مقادیر در نظر گرفته شده برای padding، مقدار valid می‌باشد که در چنین حالتی ردیف یا ستونی به اطراف تصویر اضافه نمی‌شود و اندازه نقشه ویژگی خروجی کوچک‌تر از تصویر ورودی است، از این مقدار زمانی استفاده می‌شود که بخواهیم اندازه نقشه ویژگی خروجی را کاهش دهیم تا تعداد پارامترها در مدل کاهش یابد و کارایی محاسباتی آن بهبود یابد.

برای یک بعدی کردن ورودی چندبعدی از لایه Flatten استفاده می‌شود که به طور معمول در انتقال نقشه‌های ویژگی استخراج شده از لایه‌های کانولوشن و Max Pooling به لایه کاملا متصل از این لایه استفاده می‌شود، در واقع با این کار یک ماتریس مسطح شده به عنوان ورودی به یک لایه کاملا متصل داده می‌شود تا بتواند کار طبقه‌بندی تصویر را انجام دهد. برای ایجاد یک لایه از نوع کاملا متصل از Dense استفاده می‌کنیم که نیاز است به ترتیب اندازه بردار خروجی و نوع تابع فعال‌سازی مورد استفاده (activation) را برای ایجاد لایه مشخص کنیم.

حال با توجه به توضیحات داده شده، لایه‌های مدل پیشنهادی را به ترتیب ایجاد می‌کنیم. تعداد فیلترها و اندازه فیلترهای مورد استفاده در هر لایه به صورت تجربی به دست آمده‌اند.

```
# add model layers
model.add (Conv2D (32, kernel_size=5, strides= (1,1), padding='valid', activation='relu', input_shape=(28,28,1)))
```

لایه 1 اولین لایه از نوع کانولوشن در معماری شبکه عصبی کانولوشن پیشنهادی است که تصویر پیش‌پردازش شده با اندازه 28×28 به عنوان ورودی به آن داده می‌شود. تعداد فیلترهای اعمال شده در این لایه 32 است و اندازه فیلتر کانولوشن 5×5 است که با توجه به اندازه گام 1، فیلتر به صورت پیکسل به پیکسل روی تصویر ورودی حرکت می‌کند. تابع فعال‌سازی مورد استفاده هم از نوع ReLu می‌باشد.

```
model.add (MaxPool2D (pool_size=(2,2), strides= (2,2), padding='valid'))
```

لایه 2 اولین لایه از نوع Max Pooling است. اندازه پولینگ

اعمال شده روی تصویر 2×2 است، مقدارلایه‌گذاری صفر و اندازه گام حرکت فیلتر روی تصویر 2 است.

model.add (Conv2D (32, kernel_size=5, strides= (1,1), padding='valid', activation='relu'))

لایه 3 نیز از نوع کانولوشن می‌باشد.

model.add (MaxPool2D (pool_size=(2,2), strides= (2,2), padding='valid'))

لایه 4 نیز از نوع Max Pooling می‌باشد.

model.add (Conv2D (64, kernel_size=4, strides= (1,1), padding='valid'))

لایه 5 سومین لایه کانولوشن است که تعداد فیلترها 64، اندازه فیلتر 4×4 و اندازه گام 1 است. این لایه تابع فعال‌سازی ReLu را ندارد.

model.add (Flatten())

برای یک بعدی کردن ورودی چندبعدی از لایه Flatten استفاده شده است.

model.add (Dense (256, activation='relu'))

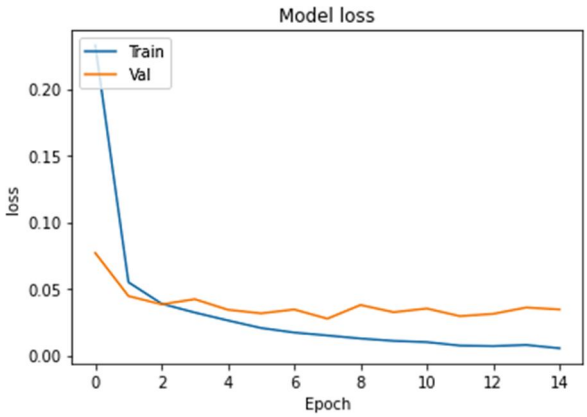
لایه 6 یک لایه کاملا متصل است که مقدار این لایه یک بردار یک بعدی به اندازه 64 را می‌گیرد و یک بردار یک بعدی با اندازه 256 را تولید می‌کند. این لایه همچنین تابع فعال‌سازی ReLu را دارد.

model.add (Dense (10, activation=' softmax '))

لایه 7 آخرین لایه شبکه است که از نوع لایه کاملا متصل است. اندازه خروجی تولید شده در این لایه 10 می‌باشد که یک بردار به اندازه 10، به طوری که هر یک از ده عدد مربوط به مقدار یک کلاس است (همانند ده دسته مجموعه داده MNIST) را داریم. تابع فعال‌سازی مورد استفاده هم از نوع softmax می‌باشد که به طور معمول در لایه خروجی از این تابع فعال‌سازی استفاده می‌شود و کار طبقه‌بندی را برای ما انجام می‌دهد.

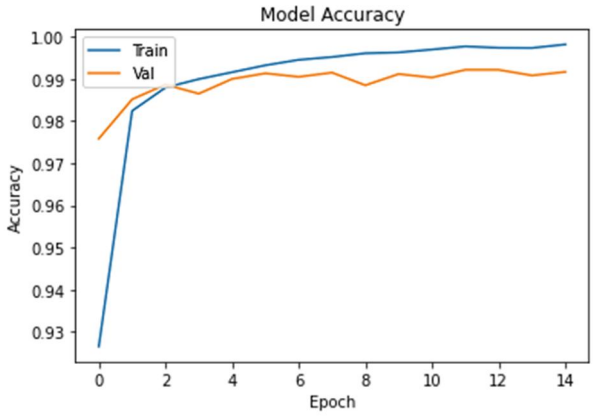
مرحله سوم آموزش و ارزیابی مدل است. آموزش CNN نیاز به محاسبه مشتقات ضرر مربوط به پارامترهای شبکه دارد. در کار حاضر برای محاسبه مشتقات از الگوریتم back propagation و برای تنظیم وزن اتصال بین سلول‌های عصبی از الگوریتم بهینه‌سازی Stochastic Gradient Descent (SGD) (گرادیان کاهشی تصادفی) استفاده شده است تا خطا به حداقل مقدار برسد یا بعد از چندین دوره متوقف شود. در آخر، می‌توان آموزش CNN را با تهیه داده‌های آموزش، مدل ساخته شده و دسته فعلی داده‌ها، شروع کرد.

در کار حاضر برای آموزش و اعتبارسنجی مدل پیشنهادی از 60000 مثال آموزشی موجود در مجموعه داده استفاده شده است که 10 درصد داده‌های آموزشی برای اعتبارسنجی مدل در نظر گرفته شده‌اند. هنگام آموزش CNN، فقط داده‌های مشخص شده برای آموزش در به حداقل رساندن خطای CNN نقش دارند. الگوریتم از داده‌های آموزشی برای پاس جلو و عقب استفاده می‌کند. همچنین این الگوریتم از داده‌های اعتبارسنجی استفاده می‌کند تا ببیند CNN چگونه به داده‌های جدید پاسخ می‌دهد. بنابراین شبکه فقط از طریق عبور به جلو تغذیه می‌شود. پس از آن، لایه‌های هزینه یا هدف (هزینه ورود به سیستم) و خطا (هزینه طبقه‌بندی) فراخوانی خواهند شد که پس از تکمیل هر دوره، تصویری از همگرایی آموزش و اعتبارسنجی را ارائه می‌دهند. در پایان، CNN آموزش دیده را ذخیره می‌کنیم و برای مرحله آزمایش آماده می‌شویم. در طول مرحله آموزش CNN، هر دوره حداکثر دو طرح (خطا و دقت) ایجاد می‌شود که در شکل‌های (8) و (9) نشان داده شده است.



شکل (8): کاهش خطا در طول آموزش

در شکل (8)، منحنی آبی رنگ نشان‌دهنده خطای آموزش و نارنجی نشان‌دهنده خطای اعتبارسنجی مدل CNN پیشنهادی در حین آموزش است.

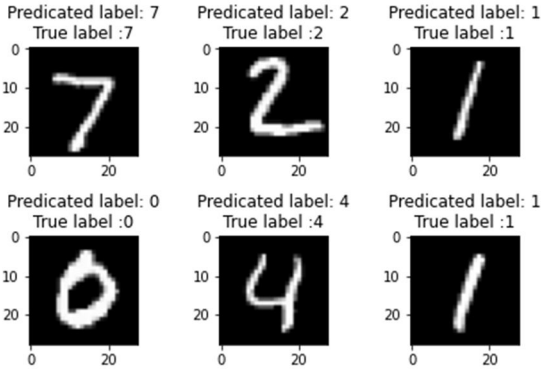


شکل (9): افزایش دقت در طول آموزش

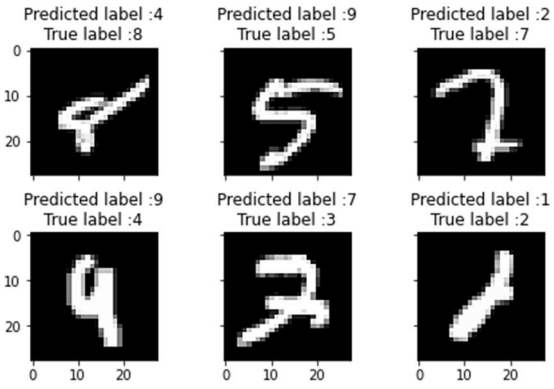
در شکل (9)، منحنی آبی نشان‌دهنده دقت آموزش و منحنی نارنجی نشان‌دهنده دقت اعتبارسنجی مدل CNN پیشنهادی در حین آموزش است. نکته حائز اهمیت در این قسمت این است که اگر مقدار این دو منحنی به هم نزدیک باشد و در واقع با هم متناسب باشند می‌توان نتیجه گرفت که مشکل بیش‌برازش یا overfit رخ نداده است.

مرحله چهارم ارزیابی و پیش‌بینی مدل پیشنهادی است که با استفاده از داده‌های آزمایشی، می‌توان مدل خود را ارزیابی کرد و دقت مدل پیشنهادی را روی مجوعه داده‌های آزمایشی به دست آورد. در جدول (1) تعداد نمونه‌های موجود در مجموعه آزمون، تعداد نمونه‌هایی که به صورت درست و نادرست طبقه‌بندی شده‌اند، دقت و خطا برای ارقام 0 تا 9 مشخص شده است.

در شکل‌های (10) و (11) چند نمونه از خروجی‌های طبقه‌بندی شده با استفاده از مدل پیشنهادی در حین آزمایش نشان داده شده است.



شکل (10): برخی از خروجی‌های درست تشخیص داده شده



شکل (11): برخی از خروجی‌های اشتباه شناخته شده

جدول (1): خلاصه‌ای از عملکرد مدل برای ارقام 0 تا 9

رقم	تعداد نمونه‌های موجود در مجموعه آزمون	تعداد نمونه‌ها با طبقه‌بندی درست	تعداد نمونه‌ها با طبقه‌بندی نادرست	دقت (%)	خطا (%)
0	980	978	2	99/79	0/21
1	1135	1134	1	99/91	0/09
2	1032	1027	5	99/51	0/49
3	1010	1009	1	99/90	0/10
4	982	973	9	99/08	0/92
5	892	874	18	97/98	2/02
6	958	950	8	99/16	0/84
7	1028	1021	7	99/31	0/69
8	974	961	13	98/66	1/34
9	1009	1003	6	99/40	0/60
کل	10000	9930	70	99/30	0/70

با افزایش مجموعه آموزش و افزایش تعداد الگوهای استاندارد می‌توان اکثر الگوها را حل کرد. علاوه بر این، از دست رفتن پیکسل‌های ناشی از فشرده‌سازی تصویر و مشکلات وضوح تصویر نیز از دلایل طبقه‌بندی نادرست است.

مرحله پنجم و آخر ذخیره مدل در حافظه برای استفاده مجدد است. ما می‌توانیم مدل آموزش دیده را ذخیره کنیم. از این رو مدل ذخیره شده می‌تواند در آینده مجدد مورد استفاده قرار گیرد یا به راحتی به سایر محیطها منتقل شود.

5. نتایج

از بین 10000 نمونه موجود برای آزمایش در مجموعه داده MNIST، مدل پیشنهادی فقط 70 رقم را به صورت نادرست طبقه‌بندی کرده است. نتایج برای چنین مدل ساده‌ای با آموزش GPU و زمان آموزش کم همان‌طور که در زیر نشان داده شده است بسیار خوب است. مقدار دقت (accuracy) و خطا (Error rate) مدل پیشنهادی در زیر آمده است.

313/313 - 1s - loss: 0.0319 - accuracy: 0.9930
accuracy: 99.29999709129333%
Error rate: 0.70%
Loss: 0.031872689723968506

در جداول (2) و (3) نتایج آزمایش روی مدل پیشنهادی با تعداد دوره و اندازه دسته‌های متفاوت نشان داده شده است.

جدول (2): نتایج آزمایش با تعداد دوره‌های مختلف در فرآیند آموزش

Accuracy	تعداد دوره‌ها	اندازه دسته
99/%16	8	40
99/%26	12	40
99/%30	15	40

جدول (3): نتایج آزمایش با اندازه دسته‌های مختلف در فرآیند آموزش

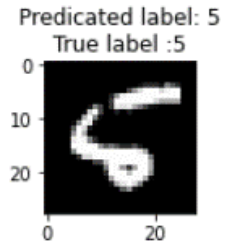
Accuracy	تعداد دوره‌ها	اندازه دسته
99/%26	15	32
99/%30	15	40
99/%23	15	64

عملکرد مدل CNN به طور کامل به بهینه‌ساز شبکه بستگی دارد. کاهش اتلاف داده‌ها، دقت بیشتر و زمان همگرایی کمتر از دلایل اساسی استفاده از بهینه‌سازها در یک شبکه است. بهینه‌سازهای مختلفی وجود دارد. SGD بهینه‌سازی است که در یادگیری عمیق بیشتر مورد استفاده قرار می‌گیرد و سرعت بالاتری دارد. می‌توان از افزونگی محاسباتی با استفاده از بهینه‌ساز SGD جلوگیری کرد. با افزودن پارامتر حرکت[4] در الگوریتم SGD، سرعت بهینه‌سازی بیشتر می‌شود که در کار حاضر نیز برای رسیدن به بالاترین دقت از این پارامتر استفاده شده است. برآورد لحظه‌ای تطبیقی به عنوان بهینه‌ساز Adam شناخته می‌شود که یک نسخه بهبود یافته از الگوریتم گرادیان تصادفی است. بهینه‌ساز Adam کارآمد است و حافظه کمتری مصرف می‌کند [17]. نتایج استفاده از بهینه‌سازهای مختلف در مدل پیشنهادی در جدول (4) نشان داده شده است، این مشاهده آزمایشی تایید می‌کند که بهینه‌ساز SGD بهترین دقت را برای مدل پیشنهادی ارائه می‌دهد و همچنین می‌توان تاثیر پارامتر momentum را در افزایش دقت مشاهده کرد.

جدول (4): دقت مدل پیشنهادی با بهینه‌سازهای مختلف

Adam	SGD (Scottish Gradient Descent)	SGDM (Scottish Gradient Descent with Momentum)
97/%75	99/%02	**99/%30**

در مجموعه داده‌ها ارقامی وجود دارد که دست خط‌های خوبی نیستند اما مدل پیشنهادی قادر به طبقه‌بندی صحیح آنها خواهد بود. به عنوان مثال، مدل پیشنهادی تصویر شکل (12) را به عنوان «5» و به صورت درست طبقه‌بندی می‌کند.



شکل (12): تشخیص صحیح خط بد

دقت آزمون 99/30 درصد نشان می‌دهد که مدل برای پیش‌بینی به خوبی آموزش دیده است. اندازه مجموعه آموزش بر دقت تاثیر می‌گذارد و دقت با افزایش تعداد داده‌ها افزایش می‌یابد. هرچه اطلاعات بیشتری در مجموعه آموزش وجود داشته باشد، تاثیر خطای آموزش و خطای آزمایش کمترین میزان را دارد و در نهایت می‌توان دقت را بهبود بخشید.

6. نتیجه‌گیری

1.6. مقایسه روش پیشنهادی با سایر روش‌ها

شبکه عصبی کانولوشن (CNN) و ماشین بردار پشتیان (SVM)

[4] Momentum

از جمله روش‌هایی هستند که برای تشخیص ارقام دست‌نویس مورد استفاده قرار می‌گیرند. در جدول (5) دقت نهایی طبقه‌بندی روش پیشنهادی و سایر روش‌های موجود قرار داده شده است.

جدول (5): مقایسه دقت بین روش پیشنهادی و سایر روش‌های موجود

روش	مجموعه داده	دقت (٪)	منبع
CNN	MNIST	98/85	[7]
CNN	MNIST	99/21	[8]
CNN	MNIST	95/7	[10]
CNN+DL4J	MNIST	99/21	[11]
CNN	MNIST	98	[13]
Histogram feature extraction+ SVM	MNIST	95/60	[14]
CNN+SVM	MNIST	99/28	[15]
CNN+SVM	MNIST	94/40	[18]
SVM	MNIST	96/93	[19]
HOG+ SVM	MNIST	97/25	[20]
DCT+ SVM	MNIST	98/88	[21]
روش پیشنهادی	MNIST	**99/30**	-

در روش پیشنهادی از شبکه عصبی کانولوشن برای فرآیند استخراج ویژگی‌های تصویر و طبقه‌بندی استفاده شده است و دقت نهایی طبقه‌بندی روش پیشنهادی 99/30% می‌باشد که نسبت به کارهای گذشته دقت بهتری دارد و برای طبقه‌بندی ارقام دست‌نویس موفق‌تر عمل کرده است. در جدول (6) مقایسه دقیق بین دو روش پیشنهادی و KNN انجام شده است و در این جدول به مزایا و معایب روش پیشنهادی در مقایسه با KNN پرداخته شده است.

جدول (6): مقایسه بین روش پیشنهادی و KNN

روش	دقت (%)	نرخ خطا (%)	مزایا	معایب
KNN	98/88	1/12	عدم نیاز به داده‌های آموزش	طبقه‌بندی کند نیاز به حافظه بالا انتخاب k
روش پیشنهادی	99/30	0/7	دقت بالا در کلاس‌بندی	محاسبات زیاد

بر اساس نتایج ارائه شده در جداول (5) و (6) می‌توان گفت در روش پیشنهادی با توجه به اهمیت طبقه‌بندی اعداد، می‌توان از معایب آن با توجه به دقت بالا صرفنظر کرد و آن را برای طبقه‌بندی استفاده کرد. از طرف دیگر، با توجه به نزدیک بودن دقت روش پیشنهادی و روش مرجع [15] در جدول (5)، در ادامه به مقایسه این دو روش پرداخته شده است. در رابطه با حجم داده‌ای مورد استفاده، هر دو روش از مجموعه داده MNIST برای آموزش و آزمایش استفاده کرده‌اند. در روش پیشنهادی برای آموزش و اعتبارسنجی مدل، از 60000 مثال آموزشی موجود در مجموعه داده MNIST استفاده شده است که 10 درصد داده‌های آموزشی نیز برای اعتبارسنجی مدل در نظر گرفته شده‌اند و برای ارزیابی و پیش‌بینی مدل نیز از 10000 مثال آزمایشی مجموعه داده MNIST استفاده شده است، دقت 99/79 از آزمایش مدل روی مجموعه داده‌های آموزشی و دقت 99/30 از آزمایش روی مجموعه داده‌های آزمایشی به دست آمده است. در مرجع [15] نیز از مجموعه داده MNIST استفاده شده است و دقت 99/28 از آزمایش روش روی مجموعه داده‌های آموزشی و دقت 98/95 از آزمایش روی مجموعه داده‌های آزمایشی به دست آمده است. در جدول (7) نیز خلاصه‌ای از عملکرد دو مدل آمده است.

جدول (7): مقایسه بین روش پیشنهادی و روش مرجع [15]

مدل مرجع [15]		مدل پیشنهادی	
آزمون	آموزش	آزمون	آموزش
98/95	99/28	**99/30**	**99/79**

پس در شرایط مشابه و آزمایش مدل‌ها روی مجموعه داده‌های آموزشی و آزمایشی مجموعه داده MNIST دقت مدل پیشنهادی از دقت مرجع [15] بیشتر است و روش پیشنهادی عملکرد بهتری داشته است. نکته دیگری که درباره روش پیشنهادی می‌توان ذکر کرد این است که مدل مورد نظر از محاسبات کارآمد CPU و GPU پشتیبانی می‌کند و می‌توان مدل را روی هر دو اجرا کرد. البته زمانی که روی GPU اجرا می‌شود چون سرعت بالایی در آموزش داده‌ها دارد زمان آموزش مدل کم می‌شود و سرعت کار بالاتر می‌رود.

2.6. کارهای آینده

استفاده از روش‌های مبتنی بر شبکه عصبی، به خصوص شبکه عصبی کانولوشنی کاربردهای فراوانی در پردازش تصاویر داشته است [22]- [24]. در اینجا نیز مدلی نشان داده شده است که می‌تواند ارقام دست‌نویس را تشخیص دهد. در آینده می‌توان آن را برای شناسایی شخصیت و دست‌خط شخص در زمان واقعی گسترش داد. شناسایی رقم دست‌نویس اولین قدم برای دستیابی به حوزه وسیع هوش مصنوعی و چشم‌انداز رایانه است. همان‌طور که از نتایج آزمایش دیده می‌شود، ثابت می‌شود شبکه عصبی کانولوشن به مراتب بهتر از سایر طبقه‌بندی‌کننده‌ها است. نتایج این کار را می‌توان با لایه‌های کانولوشن بیشتر و تعداد بیشتری از نورون‌های پنهان دقیق‌تر کرد، البته لزوما افزایش تعداد لایه‌ها منجر به بهبود یادگیری نمی‌شود و در بعضی از موارد ممکن است افزایش تعداد لایه‌ها منجر به افزایش حجم محاسبات و زمان یادگیری شود، همچنین می‌توان از ترکیب شبکه عصبی کانولوشن با روش‌های دیگر استفاده کرد و نتایج دقیق‌تری را به دست آورد. تشخیص رقم یک نمونه اولیه عالی برای یادگیری در مورد شبکه‌های عصبی است و راهی عالی برای توسعه رویکردهای پیشرفته‌تر یادگیری عمیق است. در آینده می‌توان یک سیستم شناسایی رقمی دست نوشته شده در زمان واقعی را توسعه داد.

تعارض منافع: نویسندگان اعلام می‌کنند که هیچ تعارض منافعی ندارند.

مراجع

[1] S. Ahlawat, A. Choudhary, A. Nayyar, S. Singh, and B. Yoon, “Improved handwritten digit recognition using convolutional neural networks (CNN),” Sensors, vol. 20, no. 12, p. 3344, 2020, doi: 10.3390/s20123344.

[2] M.A. Hossain and M.M. Ali, “Recognition of handwritten digit using convolutional neural network (CNN),” Global J. Comput. Sci. Technol., vol. 19, no. 2, pp. 27-33, 2019.

[3] N. Jain, K. Rahul, I. Khamaru. A.K. Jha, and A. Ghosh, “HandWritten Digit Recognition using Convolutional Neural Network (CNN),” Int. J. Innov. Adv. Comput. Sci., vol. 6, no. 5, 2017.

[4] H.A. Alwzwazy, H.M. Albehadili, Y.S. Alwan, and N.E. Islam, “Handwritten digit recognition using convolutional neural networks,” Int. J. Innov. Res. Comput. Commun. Eng., vol. 4, no. 2, pp. 1101-1106, 2016.

[5] E. Kussul and T. Baidyk, “Improved method of handwritten digit recognition tested on MNIST database,” Image Vis. Comput., vol. 22, no. 12, pp. 971-981, 2004, doi: 10.1016/j.imavis.2004.03.008.

[6] H. Wu, “CNN-Based Recognition of Handwritten Digits in MNIST Database,” Research School of Computer Science. The Australia National University, Canberra, 2018.

[7] R. Jana and S. Bhattacharyya, “Character recognition from handwritten image using convolutional neural networks,” in Recent Trends in Signal and Image Processing: Proceedings of ISSIP 2018, 2019: Springer, pp. 23-30, doi: 10.1007/978-981-13-6783-0_3.

[8] F. Siddique, S. Sakib and M.A.B. Siddique, “Recognition of Handwritten Digit using Convolutional Neural Network in Python with Tensorflow and Comparison of Performance for Various Hidden Layers,” in 5th Int. Conf. Adv. Electr. Eng. (ICAEE), Dhaka, Bangladesh, 2019, pp. 541-546, doi: 10.1109/ICAEE48663.2019.8975496.

[9] M.M. Abu Ghosh and A.Y. Maghari, “A Comparative Study on Handwriting Digit Recognition Using Neural Networks,” in Int. Conf. Promising Electron. Technol. (ICPET), Deir El-Balah, Palestine, 2017, pp. 77-81, doi: 10.1109/ICPET.2017.20.

[10] D.-Y. Ge, X.-F. Yao, W.-J. Xiang, X.-J. Wen and E.-C. Liu, “Design of High Accuracy Detector for MNIST Handwritten Digit Recognition Based on Convolutional Neural Network,” in 12th Int. Conf. Intell. Comput. Technol. Autom. (ICICTA), Xiangtan, China, 2019, pp. 658-662, doi:

10.1109/ICICTA49267.2019.00145.

[11] S. Ali, Z. Shaukat, M. Azeem, Z. Sakhawat, T. Mahmood, and K. ur Rehman, "An efficient and improved scheme for handwritten digit recognition based on convolutional neural network," SN Appl. Sci., vol. 1, no. 9, p. 1125, 2019, doi: 10.1007/s42452-019-1161-5.

[12] A.-M. Saeed, "Intelligent handwritten digit recognition using artificial neural network," Int. J. Eng. Res. Appl., vol. 5, no. 5, pp. 46-51, 2015.

[13] M. Ramprasath, M.V. Anand, and S. Hariharan, "Image classification using convolutional neural networks," Int. J. Pure Appl. Math., vol. 119, no. 17, pp. 1307-1319, 2018.

[14] E. Tuba, M. Tuba, and D. Simian, "Handwritten Digit Recognition by Support Vector Machine Optimized by Bat Algorithm," in 24th Conf. Comput. Graph. Visual. Comput. Vis., 2016, pp. 369-376.

[15] S. Ahlawat and A. Choudhary, "Hybrid CNN-SVM Classifier for Handwritten Digit Recognition," Proc. Comput. Sci., vol. 167, pp. 2554-2560, 2020, doi: 10.1016/j.procs.2020.03.309

[16] Y. LeCun, L. Bottou, Y. Bengio, and P. Haffner, "Gradient-based learning applied to document recognition," Proc. IEEE, vol. 86, no. 11, pp. 2278-2324, 1998.

[17] A. Biswas and M.S. Islam, "An efficient CNN model for automated digital handwritten digit classification," J. Inf. Syst. Eng. Business Intell., vol. 7, no. 1, pp. 42-55, 2021.

[18] X.-X. Niu and C.Y. Suen, "A novel hybrid CNN-SVM classifier for recognizing handwritten digits," Pattern Recognit., vol. 45, no. 4, pp. 1318-1325, 2012, doi: 10.1016/j.patcog.2011.09.021.

[19] C. Kaensar, "A Comparative Study on Handwriting Digit Recognition Classifier Using Neural Network, Support Vector Machine and K- Nearest Neighbor," in 9th Int. Conf. Comput. Inf. Technol. (AISC), 2013, pp. 155-163.

[20] R. Ebrahimzadeh and M. Jampour, "Efficient handwritten digit recognition based on histogram of oriented gradients and SVM," Int. J. Comput. Appl., vol. 104, no. 9, 2014.

[21] B. El Qacimy, M.A. Kerroum, and A. Hammouch, "Feature extraction based on DCT for handwritten digit recognition," Int. J. Comput. Sci. Issues (IJCSI), vol. 11, no. 6, p. 27, 2014.

[22] M. Sarchahi and E. Mahdipour, "Using ensemble deep learning to improve the accuracy of CT-Scan lung image detection of COVID-19 patients," Soft Comput. J., vol. 13, no. 1, pp. 20-39, 2024, doi: 10.22052/scj.2023.253142.1158 [In Persian].

[23] M. Eftekharian and A. Nodehi, "Breast Cancer Diagnosis and Classification Improvement based on Deep Learning and image Processing methods," Soft Comput. J., vol. 12, no .1, pp. 22-26, 2023, doi: 10.22052/scj.2023.246416.1067.

[24] M. Mousavi, S. Hosseini, and M.R. Omidi, "Improved Deep Neural Network Algorithm for Covid-19 Detection in Internet of Things," Soft Comput. J., vol. 12, no. 2, pp. 54-71, 2024, doi: 10.22052/scj.2023.248686.1117 [In Persian].