دانشگاه صنعتی امیرکبیر
(پلی‌تکنیک تهران)
دانشکده مهندسی برق

گزارش تمرین اول

شبکه‌های عصبی پیچشی

نگارش
محمدرضا شهرستانی ۴۰۳۱۱۵۰۸۳
حسین خموشی ۴۰۳۱۱۵۰۸۹

استاد
دکتر شریفیان

فروردین ۱۴۰۴

چکیده

این تمرین به طراحی و آموزش شـبکه‌های عصـبی پیچشـی می‌پردازد. در ابتدا چهار dataset قرار داده شده بود. ما ویژگی‌های هر چهار dataset را، اعم از نوع تصاویر، پراکندگی و توزیع داده‌ها در دسته‌های مختلف، نحوه برچسب‌گذاری، داده‌های خراب و... را بررسی کردیم و خلاصه آن را ارائه دادیم. با توجه به نکاتی که از بررسی چهار dataset به دست آوردیم، دو dataset که امکان ترکیب شدن دارند را انتخاب کردیم. بعد از انتخاب dataset های مناسب، یک setup انجام دادیم؛ به این صورت که تصاویر را هم‌اندازه کردیم؛ نام‌گذاری فایل‌ها را برای راحتی کار تغییر دادیم و یک پوشه درست کردیم و نتیجه را در آن قرار دادیم. در واقع مرحله preprocess را که مرحله بسـیار مهمی اسـت انجام دادیم. دیتا را به سـه بخش train، test و validation تقسیم کردیم. یک شبکه عصبی پیچشی ساختیم و آن را train کردیم سپس به ارزیابی آن پرداختیم. سپس با توجه به متن تمرین تغییراتی در فیلترها اعمال کردیم، padding اضافه کردیم، stride را تغییر دادیم، لایه اضافه کردیم، فیلترها را کاهش دادیم، سایز max pooling را افزایش دادیم و از avg max pooling استفاده کردیم و تغییرات زیادی اعمال کردیم و به صورت جزئی به این پرداختیم که با تغییر هر کدام از عناوینی که اشاره کردیم چه تغییری در خروجی حاصل شد. به نتیجه مثبتی منجر شـد یا بالعکس؟ تغییراتی دیگری نیز مانند اسـتفاده از activation function های دیگر و تغییر لایه fully connected، تغییر بهینه‌سـاز و ... را هم اعمال کردیم و نتایج را به صـورت جزئی مورد بررسی قرار دادیم. سعی شده مطالب به صورت روان توضیح داده شود و حتی‌الامکان از تصاویر، جدول، نمودار و... برای نمایش خروجی‌ها، مقایسه‌ها و مفاهیم استفاده شود.

لینک کدهای گزارش:

https://colab.research.google.com/drive/1exZxLRBveWJA1QPUpf55X3r0d6XD4iF5?usp=sharing

واژه‌های کلیدی:

پیش‌پردازش داده‌ها، یادگیری عمیق، تشخیص احساسات، شبکه عصبی پیچشی، پردازش تصویر

فهرست مطالب

صفحه

۱) شناخت پایگاه داده ۴

۱-۱ انتخاب پایگاه داده ۴

۱-۲ بررسی مقاله پایگاه داده ۱۰

۲) پیش‌پردازش داده‌ها(Data Preprocessing) ۱۴

۲-۱ آماده‌سازی داده‌ها(Data Preparation) ۱۴

۲-۲ تقسیم‌بندی داده‌ها(Data Splitting) ۱۹

۳) ساخت شبکه عصبی ۱۹

۳-۱ شبکه CNN ساده ۱۹

۳-۲ انتخاب معماری اولیه ۲۱

۳-۲-۱ تغییر اندازه فیلترها ۲۱

۳-۲-۲ تغییر Padding ۲۱

۳-۲-۳ تغییر Stride ۲۱

۳-۲-۴ تعداد لایه‌ها و فیلترها ۲۱

۳-۲-۵ تغییر Pooling ۲۲

۳-۲-۶ توابع فعال‌سازی ۲۲

۳-۲-۷ تغییر Fully Connected Layer ۲۲

۳-۲-۸ تحلیل و انتخاب معماری نهایی ۲۲

۴) آموزش شبکه عصبی ۲۴

۴-۱ بهینه‌سازی مناسب ۲۴

۴-۱-۱ تغییر نرخ یادگیری ۲۴

۴-۱-۲ استفاده از scheduler در تغییر نرخ یادگیری ۲۵

۴-۱-۳ امتحان کردن ضرایب مختلف رگولاسیون ۲۶

۴-۱-۴ تغییر و مقایسه با سایر تابع‌های بهینه‌ساز ۲۷

۴-۲ لایه‌های کمکی ۲۹

۴-۲-۱ اضافه کردن لایه BatchNorm ۲۹

۴-۲-۲ اضافه کردن لایه Dropout ۲۹

۴-۲-۳ بررسی تاثیر همزمان دو لایه Batchnorm و Dropout ۳۱

۴-۳ توقف مناسب آموزش ۳۳

۴-۳-۱ تغییر epoch ۳۳

۴-۳-۲ استفاده از Early Stopping ۳۴
۴-۳-۳ استفاده از ModelCheckpoint (ذخیره بهترین مدل بر اساس دقت اعتبارسنجی) ۳۵
۴-۳-۴ مقایسه عملکرد سه روش ۳۶
۴-۴ تنظیم مدل ۳۷
۴-۵ انتخاب نهایی مدل ۴۰
۴-۶ انتقال یادگیری ۴۰
۴-۷ تاثیر پایگاه داده بر آموزش ۴۲
۵) استفاده از مدل شبکه عصبی ۴۲
۵-۱ دقت بر روی سایر پایگاه داده ۴۲
نتایج روی affectnet ۴۲
نتایج روی JAFFE ۴۳
Abstract ۴۴

۱) شناخت پایگاه داده

۱-۱ انتخاب پایگاه داده

چهار dataset در حوزه تشـخیص احسـاسـات قرار داده شـده بود که به بررسـی هر کدام می‌پردازیم در نهایت دو dataset را انتخاب می‌کنیم.

الف) dataset اول با نام "EmotionDetection_happy_or_sad" اســت که در لینک زیر قرار دارد:

https://www.kaggle.com/datasets/aravindanr22052001/emotiondetection-happy-or-sad

نمونه تصاویر دیتاست EmotionDetection_happy_or_sad

این dataset شامل تصاویر طبقه‌بندی شده به دو احساس "خوشحال" و "ناراحت" می‌باشد. ویژگی‌های دیگر این dataset به شرح زیر است:

- **نوع تصاویر**: تصاویر موجود در این dataset، تصاویر رنگی از چهره افراد هستند که احساسات "خوشحال" و "ناراحت" را نمایش می‌دهند.
- **اندازه تصاویر**: تصاویر دارای size های مختلفی هستند. size متداول استفاده شده 48 * 48 پیکسل است.
- **پراکندگی و توزیع داده‌ها در دسته‌های مختلف**: تقریبا ۵۵ درصد تصاویر در دسته happy و ۴۵ درصد در دسته sad قرار دارند.
- **حجم داده‌ها**: ۶۸ مگابایت (به صورت فایل زیپ)

- **نحوه برچسب گذاری**: برچسب گذاری در این dataset به صورت پوشه‌بندی انجام شده است؛ به این صورت که پوشه happy شامل تصاویر با برچسب خوشحال است و پوشه sad شامل تصاویر با برچسب ناراحت است.
- **فرمت تصاویر**: تصاویر این dataset در فرمت‌های jpg، jpeg و png قرار دارد. عمده تصاویر در فرمت jpg هستند که اینها رایج‌ترین فرمت‌ها برای پردازش تصویر در مدل‌های یادگیری ماشین است. این فرمت‌ها فشرده‌سازی مناسبی دارند و به راحتی در کتابخانه‌های پردازش تصویر مانند OpenCV و PIL قابل استفاده هستند.

ب) دیتاست دوم با نام AffectNet در لینک زیر قرار دارد:

https://www.kaggle.com/datasets/thienkhonghoc/affectnet/data

نمونه تصاویر دیتاست AffectNet

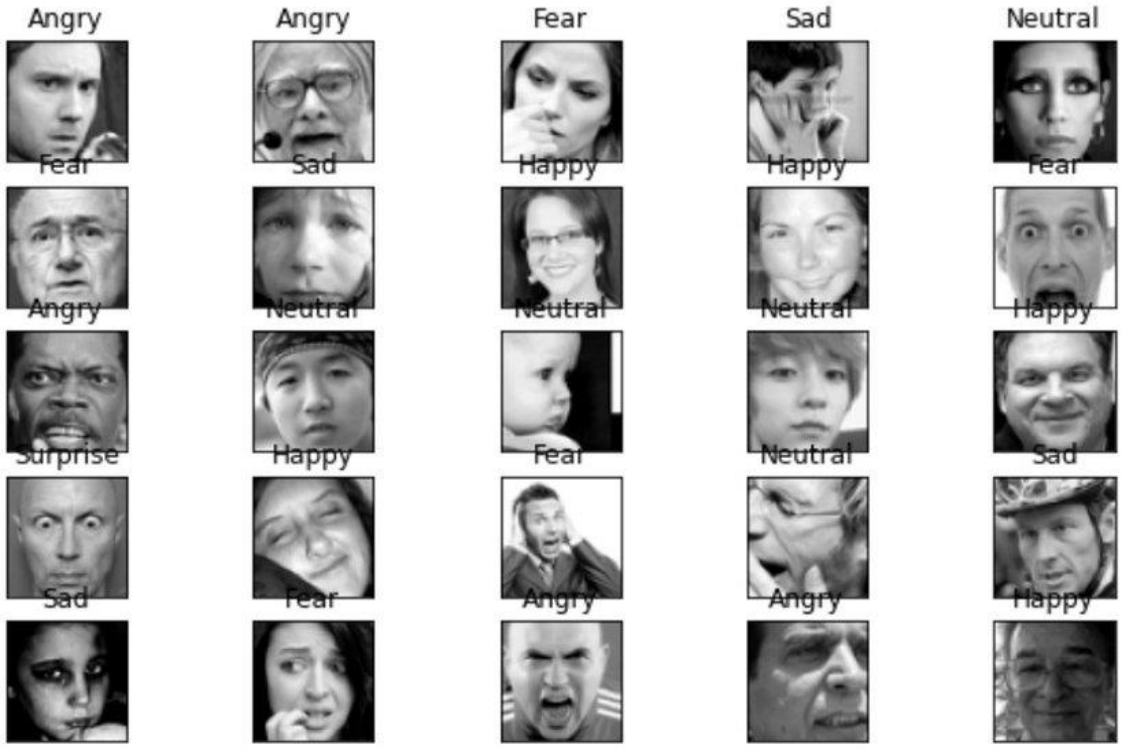
- **نوع تصاویر**: شامل تصاویر RGB رنگی از چهره‌های انسانی است.

- **اندازه تصاویر**: تصاویر معمولا در اندازه‌های متفاوتی هستند، اما common size آنها در حدود 224*224 پیکسل و بالاتر است.
- **پراکندگی و توزیع داده‌ها در دسته‌های مختلف**: این دیتاست دارای هشت کلاس احساسی اصلی است:
  - شادی
  - غم
  - خشم
  - ترس
  - تعجب
  - نفرت
  - بی تفاوتی
  - حالت ترکیبی

  توزیع داده‌ها در بین این کلاس‌ها نامتعادل است؛ برخی احساسات (مانند شادی و بی‌تفاوتی) بیشتر از بقیه تکرار شده‌اند.
- **حجم داده‌ها**: این دیتاست شامل میلیون‌ها تصویر از چهره‌های دارای احساسات مختلف است. حجم فایل zip شده این داده‌ها حدود 2GB است.
- **نحوه برچسب گذاری**: هر تصویر با یک برچسب احساسی مشخص شده است .علاوه بر احساسات دسته‌بندی‌شده، این دیتاست شامل مقادیر پیوسته برای خوشحالی (valence) و برانگیختگی (arousal) است که شدت احساسات را نشان می‌دهد.
- **داده‌های خراب یا از دست رفته**: برخی تصاویر کیفیت پایینی دارند یا چهره به درستی در آن نمایش داده نشده است.
- **فرمت تصاویر**: تصاویر معمولاً در فرمت JPG یا PNG ذخیره شده‌اند. برچسب‌ها و اطلاعات مربوط به احساسات در فایل‌های CSV همراه با مختصات نقاط کلیدی ذخیره شده‌اند.
- **موقعیت چهره و نقاط کلیدی (Facial Landmarks)**: این دیتاست دارای ۶۸ نقطه کلیدی (landmarks) برای چهره‌ها است که برای تحلیل حالت‌های چهره بسیار مفید هستند. این نقاط شامل چشم‌ها، ابروها، بینی، لب‌ها و فک هستند.

ج) دیتاست سوم با نام fer2013 در لینک زیر قرار دارد:

https://www.kaggle.com/datasets/msambare/fer2013/data



نمونه تصاویر دیتاست fer2013

- **نوع تصاویر**: شامل تصاویر چهره‌های انسانی است که به‌صورت Grayscale (سطوح خاکستری) و وضوح پایین ثبت شده‌اند. این تصاویر از چهره‌های مختلف در شرایط نوری و زاویه‌های متنوع تهیه شده‌اند.
- **اندازه تصاویر**: تمامی تصاویر دارای اندازه ثابت ۴۸×۴۸ پیکسل هستند. این تصاویر به‌صورت سیاه و سفید (Grayscale) ذخیره شده‌اند.
- **پراکندگی و توزیع داده‌ها در دسته‌های مختلف**: این دیتاست دارای هفت کلاس احساسی اصلی است:
  - خشم (angry)
  - انزجار (Disgust)
  - ترس (Fear)
  - خوشحالی (Happy)
  - غم (Sad)
  - تعجب (Surprise)

- خنثی (Neutral)

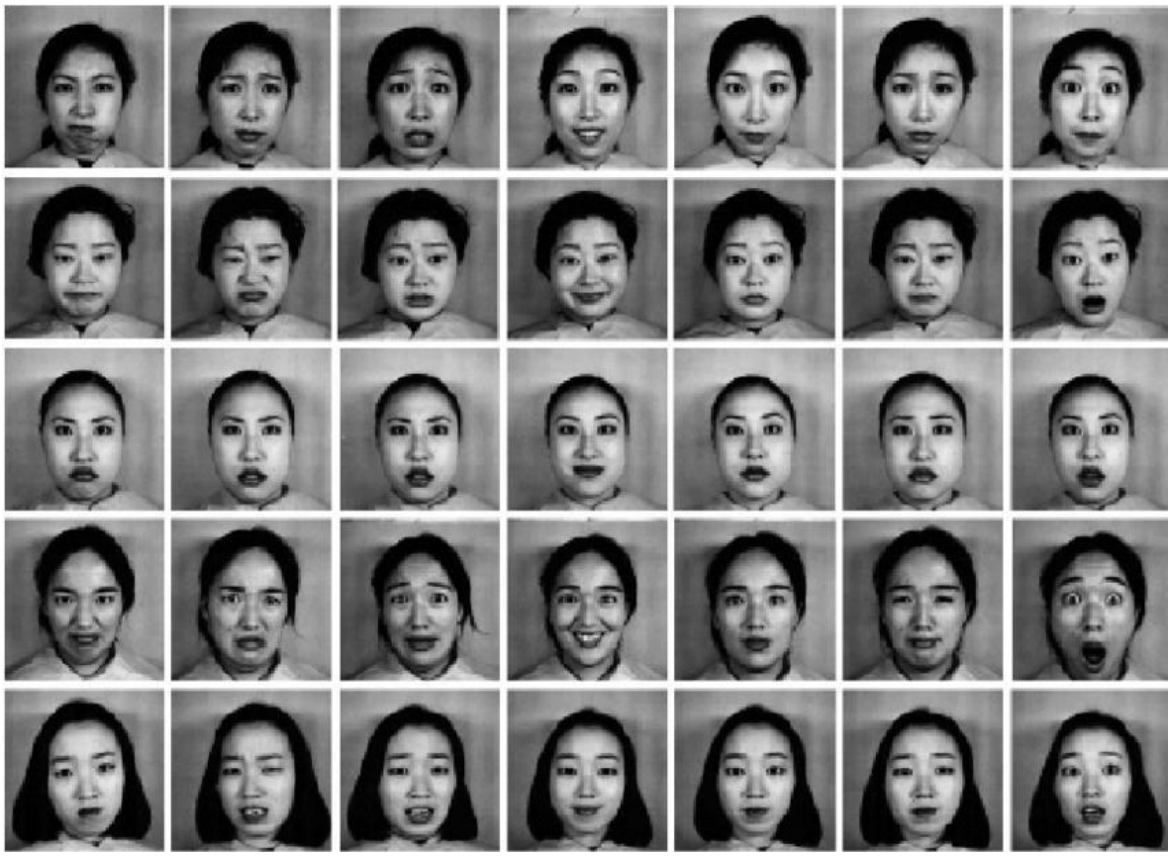
توزیع داده‌ها در بین کلاس‌ها نامتعادل اســت. برخی کلاس‌ها مانند خوشــحالی و غم دارای داده‌های بیشتری هستند، درحالی‌که کلاس انزجار کمترین داده را دارد.

- **حجم داده‌ها**: این دیتاست شامل 35,887 تصویر چهره است. حجم کلی فایل‌ها در حدود 100+ مگابایت است.
- **نحوه برچسب گذاری**: هر تصویر دارای یک عدد برچسب احساسی (از ۰ تا ۶) است که نشان‌دهنده نوع احساس چهره است. برچسب‌ها در یک فایل CSV همراه با داده‌های تصویر ذخیره شده‌اند.
- **داده‌های خراب یا از دست رفته**: برخی تصاویر نامشخص یا دارای نویز می‌باشند. به دلیل وضوح پایین(48×48) ، جزئیات دقیق چهره‌ها کمتر قابل مشاهده هستند که ممکن است در شناسایی صحیح احساسات تأثیر بگذارد.
- **فرمت تصاویر**: تمام تصاویر در این دیتاست به‌صورت آرایه‌های پیکسلی در فایل CSV ذخیره شده‌اند و نه به‌عنوان فایل‌های جداگانه. هر تصویر به‌صورت یک بردار ۴۸×۴۸ پیکسلی نمایش داده می‌شود که مقادیر آن از ۰ (سیاه) تا ۲۵۵ (سفید) متغیر است.
- **تقسیم‌بندی داده‌ها به مجموعه‌های آموزشی و آزمایشی**
  - دیتاست به دو بخش تقسیم شده است:
    - Training Set (مجموعه آموزش): شامل 28,709 تصویر 80~٪ (داده‌ها)
    - Test Set (مجموعه آزمایش): شامل 7,178 تصویر (20~٪ داده‌ها)

این تقسیم‌بندی برای آموزش مدل‌های یادگیری عمیق و ارزیابی عملکرد آن‌ها استفاده می‌شود.

د) دیتا ست چهارم با نام The Japanese Female Facial Expression (JAFFE) در لینک زیر قرار دارد:

https://zenodo.org/records/3451524



نمونه تصاویر دیتاست JAFFE

- **نوع تصاویر**: تصاویر از چهره‌های زنان ژاپنی در پس‌زمینه‌ای یکنواخت و نورپردازی کنترل‌شده گرفته شده‌اند. چهره‌ها معمولاً از زاویه روبرو گرفته شده‌اند تا تغییرات احساسی به‌وضوح دیده شوند.
- **اندازه تصاویر**: تصاویر دارای اندازه 256×256 پیکسل هستند. به‌صورت سیاه و سفید (Grayscale) ذخیره شده‌اند.
- **پراکندگی و توزیع داده‌ها در دسته‌های مختلف**: تصاویر به 7 دسته احساسی مختلف تقسیم شده‌اند:
  - خشم (Angry)
  - انزجار (Disgust)
  - ترس (Fear)
  - خوشحالی (Happy)
  - غم (Sad)
  - تعجب (Surprise)

- خنثی (Neutral)

هر دسته شامل تقریباً ۳۰ تصویر است، اما تعداد تصاویر در همه کلاس‌ها دقیقاً برابر نیست.

- **حجم داده‌ها**: این دیتاست بسیار کوچک‌تر از سایر مجموعه‌های داده‌های احساسی است. شامل 213 تصویر از 10 زن ژاپنی است.
- **نحوه برچسب گذاری**: هر تصویر دارای یک برچسب احساسی بر اساس قضاوت انسانی است. برچسب‌ها در یک فایل متنی همراه با رتبه‌بندی شدت احساسات توسط چندین داور ذخیره شده‌اند.
- **داده‌های خراب یا از دست رفته**: این دیتاست نقص فنی یا تصاویر نامفهوم ندارد زیرا در یک محیط کنترل‌شده ثبت شده است. با این حال، حجم کم داده‌ها ممکن است در یادگیری مدل‌های یادگیری عمیق محدودیت ایجاد کند.
- **فرمت تصاویر**: تصاویر در فرمت TIFF ذخیره شده‌اند. هر تصویر دارای وضوح بالا و نویز کم است، که آن را برای تحلیل دقیق احساسات چهره‌ای ایده‌آل می‌کند.
- **محدودیت‌های دیتاست**: (تنوع نژادی ندارد) فقط شامل زنان ژاپنی است. حجم داده بسیار کم است که برای آموزش مدل‌های مدرن شبکه‌های عصبی عمیق (Deep Learning) کافی نیست. محیط عکس‌برداری کنترل‌شده است و احساسات مصنوعی‌تر از احساسات واقعی در موقعیت‌های طبیعی هستند.

۱-۲ بررسی مقاله پایگاه داده

در لینک https://arxiv.org/html/2402.01355v2 مقاله با عنوان FindingEmo: An Image Dataset for Emotion Recognition in the Wild قرار دارد. در اینجا به بررسی این مقاله می‌پردازیم.

نمونه تصاویر جمع‌آوری شده برای ساخت دیتاست

۱. نحوه جمع‌آوری داده‌ها (Data Collection Process)

فاز اول: جمع‌آوری تصاویر

- از یک اسکریپر (Scrapy) سفارشی مبتنی بر Python و موتور جستجوی DuckDuckGo استفاده شده است.

- سه دسته کلیدواژه تعریف شد :
  - واژگان احساسی (مثلا happy, sad)
  - مکان‌های اجتماعی (مانند workplace)
  - گروه‌های سنی (مثل adults, seniors)

- ترکیبی تصادفی از این واژگان ساخته شد و برای هر ترکیب، نتایج جستجوی تصویری گردآوری گردید.

- نتایج اولیه فیلتر شـــدند تا دامنه‌های نامطلوب (مانند ســـایت‌های عکس اســـتوک) و اندازه تصاویر نامناسب حذف شوند.
- در مرحله اول، حدود 1,041,105 تصویر جمع‌آوری گردید.

فاز دوم

- با ا ستفاده از یک رابط وب سفار شی (Python، HTML، JavaScript) برچ سب‌گذاری انجام شد.
- 655 نفر از طریق پلتفرم Prolific به‌عنوان برچســب‌گذار انتخاب شــدند (مرد/زن، نرمال، زبان انگلیسی روان).
- برچسب‌گذاری شامل موارد زیر بود :
  - ولانس (Velance): از ۳- تا ۳+
  - آروزل (Arousal): از ۰ تا ۶ (در رابط کاربری به Intensity تغییر نام داده شــده است)
  - برچسب احساسی: از چرخ احساسات Plutchik با ۸ گروه احساسی (Emo8) و ۲۴ زیرده (Emo24)
- برچســـب‌گذارها ابتدا باید تصـــویر را به عنوان مناســـب (Keep) یا رد (Reject) ارزیابی می‌کردند. سپس، ابعاد دیگر را برای تصاویر "Keep" برچسب می‌زدند.
- برچسب‌گذاری در 51 نوبت جداگانه انجام شد، با هر نوبت شامل ۱۰-۱۵ نفر.

۲. پیش پردازش داده‌ها (Data Preprocessing)

- تصاویر بر اساس تصمیم برچسب‌گذار به Keep یا Reject تقسیم شدند .
- برای حذف نویزهای اولیه، یک CNN داخلی برای تشخیص Reject/Keep ساخته شد که با دقت 83.6% تصاویر نامناسب را فیلتر کرد.

- برای برقراری تعادل میان کلاس‌های احساسی (به‌ویژه جلوگیری از غلبه "Joy" و "Anticipation") از یک CNN برای پیش‌بینی Emo8 استفاده شد و تصاویر جدید با احتمال برچسب‌های کمتر نمایش داده شدند.
- ابعاد تصاویر برای پردازش مدل‌های مختلف به اندازه‌های استاندارد (600*800 یا 602*798) بازنمایی و نرمال‌سازی شدند .
- تعادل کلاس‌ها: الگوریتم‌های بالانسینگ اعمال شد اما همچنان کلاس‌هایی مانند "Disgust" و "Surprise" کمتر نمایان شدند .
- برچسب‌گذاری چند مرحله‌ای با کنترل کیفیت شامل تصاویر آزمون (overlap images) برای ارزیابی قابلیت اطمینان برچسب‌گذاران انجام شد.

۳. پراکندگی داده‌ها (Data Distribution)

ساختار کلی مجموعه داده:

- 25,869 تصویر در مجموعه عمومی (Public Set)
- 1,525 تصویر در مجموعه خصوصی (Private Set)
  - تصاویر خصوصی با برچسب‌های چندگانه جهت استفاده در تست‌های ویژه و بررسی سازگاری برچسب‌گذاران
- مشخصات آماری :
  - 80.9% تصاویر تک برچسب (single-label)
  - 19.1% تصاویر چندبرچسب (multi-label)

توزیع برچسب‌های احساسی (Emo8):

- بیشترین تصاویر مربوط به Joy و Anticipation

کمترین تصاویر مربوط به Disgust و Surprise (نمودارهای مرتبط در بخش A.10 ضمیمه مقاله موجود است)

آمار میانگین و انحراف معیار مقادیر ولانس، آروزل و ابهام برای هر گروه احساس:

- مثال :

- Joy: ولانس ۱,۹+، آروزل ۲,۹۶+
- Sadness: ولانس ۱,۵۷−، آروزل ۳,۴۲+ (جدول کامل در بخش A.10 ارائه شده)

۴. Ambiguity (ابهام)

- مقیاسی از ۰ تا ۶ برای سنجش دشواری برچسب‌گذاری عکس از نظر احساسی.
- تصاویر با احساسات دوگانه یا متناقض، ابهام بالایی داشتند.

۵. Deciding Factors for Emotion (عوامل تصمیم‌گیری)

- تحلیل‌هایی از اینکه کدام عناصر تصویر (زبان بدن، حالت چهره، محیط و غیره) بیشترین تأثیر را بر انتخاب برچسب احساسی داشته‌اند.
- در رابط کاربری این گزینه‌ها انتخاب‌شدنی بودند و در آنالیز نهایی استفاده شدند.

۲) پیش‌پردازش داده‌ها (Data Preprocessing)

پیش‌پردازش داده‌ها یکی از مهم‌ترین مراحل در هر پروژه یادگیری ما شین ا ست، زیرا کیفیت داده‌های ورودی تأثیر مستقیمی بر عملکرد مدل نهایی خواهد داشت. در این پروژه، فرآیند پیش‌پردازش در دو بخش زیر انجام شده است:

۲-۱ آماده‌سازی داده‌ها (Data Preparation)

الف) بررسی کلی و انجام کارهای اولیه

- با استفاده از drive.mount('/content/drive')، Google Drive به محیط Colab متصل می‌شویم تا به فایل‌های ذخیره‌شده در آن دسترسی پیدا کنیم.
- دسترسی به Kaggle api را تنظیم می‌کنیم.
- دو مجموعه داده را از Kaggle دانلود می‌کنیم. فایلهای دانلودی را unzip می‌کنیم و به پوشه‌های مربوطه خودشان یعنی emotiondetection_happy_or_sad و fer2013 انتقال می‌دهیم. سپس فایلهای زیپ را حذف می‌کنیم.

- حالا دو پوشـــه happy و sad ایجاد می‌کنیم. تمام تصـــاویر مربوطه به happy و sad از هر دو مجموعه داده emotiondetection_happy_or_sad، fer2013 را در پوشه‌های مربوطه می‌ریزیم:
  - تصاویر "happy" و "sad" از زیرپوشه‌های train و test مجموعه fer2013.
  - تصـــاویـــر "happy" و "sad" از زیـــرپـــوشـــه data مـــجـــمـــوعـــه emotiondetection_happy_or_sad.
  - سپس پوشه‌های اضافی را حذف کردیم.

نتیجه

دانلود و ادغام داده‌ها: این کد دو مجموعه داده تصویری را از Kaggle دانلود کرده و تصاویر مربوط به دو کلاس "happy" و "sad" را از هر دو مجموعه در دو پوشه مجزا (happy و sad) ادغام می‌کند.

آماده‌ســـازی: این عملیات بخشـــی از پیش‌پردازش داده‌هاســـت تا داده‌ها برای مراحل بعدی (مثل تقسیم‌بندی به مجموعه‌های آموزشی/اعتبارسنجی/آزمایشی و آموزش مدل) آماده شوند.

تمیز کردن فضای کاری: حذف فایل‌ها و پوشه‌های غیرضروری برای سازمان‌دهی بهتر.

ب) انجام مراحل Preprocess

این کد برای یافتن پراستفاده‌ترین اندازه (ابعاد) تصاویر در دو پوشه `happy` و `sad` استفاده می‌شود. در ادامه، عملکرد کد به‌صورت مختصر توضیح داده شده است:

۱. وارد کردن کتابخانه‌ها:

- `PIL.Image` برای باز کردن و پردازش تصاویر.
- `Counter` از `collections` برای شمارش تعداد تکرار هر اندازه تصویر.

۲. تعریف تابع `find_most_frequent_size`:

- این تابع مسیر یک پوشه (`folder_path`) را دریافت می‌کند.

- برای هر فایل در پوشه:

- بررسی می‌کند که فایل باشد (نه پوشه) با `os.path.isfile`.

- تصویر را با `Image.open` باز کرده و ابعاد آن (عرض و ارتفاع) را با `img.size` استخراج می‌کند.

- ابعاد به لیست `sizes` اضافه می‌شوند.

- در صورت بروز خطا (مثل فایل غیرتصویری)، خطا چاپ می‌شود.

- با استفاده از `Counter`، تعداد تکرار هر اندازه شمرده شده و شایع‌ترین اندازه با `most_common(1)` پیدا می‌شود.

- اگر اندازه‌ای پیدا شد، شایع‌ترین اندازه (به‌صورت `(width, height)`) برگردانده می‌شود؛ در غیر این صورت، `None` برگردانده می‌شود.

۳. اجرای تابع:

- تابع برای پوشه‌های `happy` و `sad` فراخوانی می‌شود.

- شایع‌ترین اندازه تصاویر در هر پوشه (`common_size_happy` و `common_size_sad`) محاسبه و چاپ می‌شود.

هدف کلی و نتیجه

- تحلیل ابعاد تصاویر: کد بررسی می‌کند که کدام اندازه (عرض و ارتفاع) در تصاویر پوشه‌های `happy` و `sad` بیشترین تکرار را دارد. خروجی کد دو مقدار `(width, height)` را برای پوشه‌های `happy` و `sad` چاپ می‌کند که نشان‌دهنده شایع‌ترین ابعاد تصاویر در هر پوشه است.

ج) ادامه فرآیند Preprocess

۱. تعریف تابع `resize_images_to_common_size`:

- این تابع دو آرگومان دریافت می‌کند:

- `folder_path`: مسیر پوشه‌ای که تصاویر در آن قرار دارند (مثل `happy` یا `sad`).

- `common_size`: اندازه هدف برای تغییر اندازه تصاویر (به‌صورت `(width, height)`).

- برای هر فایل در پوشه:

- بررسی می‌کند که فایل باشد (نه پوشه) با `os.path.isfile`.

- تصویر را با `Image.open` باز می‌کند.

- تصویر را با `img.resize(common_size)` به اندازه مشخص‌شده تغییر می‌دهد.

- تصویر تغییر اندازه داده شده را در همان مسیر فایل اصلی با `img_resized.save(file_path)` ذخیره می‌کند (فایل اصلی جایگزین می‌شود).

- پیام تأیید چاپ می‌شود (مثل: `filename resized to (width, height)`).

- در صورت بروز خطا (مثل فایل غیرتصویری)، خطا چاپ می‌شود.

۲. انتخاب اندازه مشترک:

- کد بررسی می‌کند که آیا `common_size_happy` و `common_size_sad` (شایع‌ترین اندازه‌های تصاویر در پوشه‌های `happy` و `sad` که قبلاً محاسبه شده‌اند) وجود دارند یا خیر.

- اگر هر دو وجود داشته باشند:

- اگر `common_size_happy` و `common_size_sad` یکسان باشند، همان اندازه انتخاب می‌شود.

- در غیر این صورت، به صورت پیش‌فرض `common_size_happy` به‌عنوان `common_size` انتخاب می‌شود.

- اندازه مشترک چاپ می‌شود (مثل: `Common size found: (width, height)`).

- اگر اندازه مشترکی پیدا نشود، پیام خطا چاپ می‌شود: `Could not find common size. Please check the folders`.

۳. تغییر اندازه تصاویر:

- اگر `common_size` معتبر باشد، تابع `resize_images_to_common_size` برای هر دو پوشه `happy` و `sad` فراخوانی می‌شود.

- تمام تصاویر در این پوشه‌ها به اندازه `common_size` تغییر اندازه داده می‌شوند.

هدف و نتیجه

این کد اطمینان می‌دهد که تمام تصـــاویر در پوشـــه‌های `happy` و `sad` دارای ابعاد یکســـانی (`common_size`) باشـــند. این کار برای آماده‌ســـازی داده‌ها برای مدل‌های یادگیری عمیق ضـــروری است، زیرا مدل‌ها نیاز به ورودی‌هایی با ابعاد یکسان دارند. تمام تصاویر در پوشه‌های `happy` و `sad` به یک اندازه مشـــترک (شـــایع‌ترین اندازه در `happy`) تغییر اندازه داده می‌شـــوند. در اینجا به ۴۸×۴۸ تغییر سایز یافته‌اند. فایل‌های اصلی با نسخه‌های تغییر اندازه‌یافته جایگزین می‌شوند.

د) تغییر نام داده‌ها

- در این مرحله نام فایل‌های تصـویری را به یک فرمت اسـتاندارد و منظم (با پیشـوند مشـخص و شماره‌گذاری ترتیبی) تغییر می‌دهیم تا شناسایی و مدیریت آن‌ها در مراحل بعدی آسان‌تر شود.
- **نتیجه**
- فـایـل‌هـای موجود در پوشـــــه happy بـه نـام‌هـایی مثـل happy0000001.jpg, happy0000002.jpg, ... تغییر نام می‌یابند.
- فایل‌های موجود در پوشـــه sad به نام‌هایی مثل sad0000001.jpg, sad0000002.jpg, ... تغییر نام می‌یابند.
- این کار باعث می‌شـــود فایل‌ها به‌صـــورت یکنواخت و قابل‌پیگیری برای مراحل بعدی پروژه آماده شوند.

ه) ادغام تصاویر

هدف کلی و نتیجه

- **ادغام داده‌ها** :تصـــاویر موجود در پوشـــه‌های happy و sad در یک پوشـــه واحد (dataset) جمع‌آوری می‌شوند تا برای مراحل بعدی (مثل تقسیم‌بندی داده‌ها) آماده شوند.

- **ســـازمان‌دهی و تمیز کردن** :حذف پوشـــه‌های اصـــلی برای جلوگیری از پراکندگی داده‌ها و مرتب‌سازی فضای کاری.
- تمام تصاویر از پوشه‌های happy و sad به پوشه dataset منتقل (کپی) شدند.
- پوشه‌های happy و sad حذف شدند.
- پوشه dataset شامل تمام تصاویر (با نام‌های اصلی) آماده برای پردازش‌های بعدی است.

۲-۲ تقسیم‌بندی داده‌ها (Data Splitting)

- **هدف و نتیجه**
- **تق سیم داده‌ها** : داده‌های تصـویری را به‌صـورت تصـادفی به سـه مجموعه آموزشـی (۷۰٪)، اعتبارسنجی (۱۵٪)، و آزمایشی (۱۵٪) تقسیم کردیم.
- **سازمان‌دهی داده‌ها** : فایل‌ها به زیرپوشـــه‌های مجزا منتقل می‌شـــوند تا بارگذاری و مدیریت آن‌ها در مراحل بعدی آسان‌تر باشد.
- فایل‌های موجود در پوشـــه dataset به ســـه زیرپوشـــه dataset/train, dataset/validation, و dataset/test منتقل می‌شوند.
- هر زیرپوشه شامل نسبت مشخصی از داده‌ها (۷۰٪، ۱۵٪، ۱۵٪) است.
- این ساختار برای استفاده در فرآیندهای آموزش مدل مناسب است.

۳) ساخت شبکه عصبی

۳-۱ شبکه CNN ساده

الف) معماری مدل پیاده‌سازی‌شده:

- مدل شامل دو لایه کانولوشنی (Convolutional Layer) به همراه MaxPooling است.
- پس از لایه‌های پیچشی، یک لایه Fully Connected (Dense Layer) قرار دارد.

- ساختار کلی شبکه:
  - Conv2D Layer 1
    - **تعداد فیلتر**: ۳۲
    - **اندازه فیلتر**: ۳×۳
    - **فعال‌ساز**: ReLU
    - به دنبال آن MaxPooling2D با اندازه ۲×۲
  - Conv2D Layer 2
    - **تعداد فیلتر**: ۶۴
    - **اندازه فیلتر**: ۳×۳
    - **فعال‌ساز**: ReLU
    - به دنبال آن MaxPooling2D با اندازه ۲×۲
  - Flatten Layer برای تبدیل ویژگی‌ها به بردار ۱ بعدی.
  - Fully Connected Layer
    - ۱۲۸ نورون
    - **فعال‌ساز**: ReLU
  - خروجی (Output Layer)
    - تعداد نورون‌ها بر اساس تعداد کلاس‌ها
    - **تابع فعال‌ساز**: Softmax

ب) تابع هزینه و بهینه‌ساز

- **تابع هزینه sparse_categorical_crossentropy**: مناسب برای دسته‌بندی چندکلاسه.
- **بهینه‌ساز SGD (Stochastic Gradient Descent) :** با نرخ یادگیری ۰,۰۱

ج) روند آموزش

- آموزش مدل به مدت 20 دوره (epoch) انجام شده.
- در هر دوره، دقت و خطای مربوط به داده‌های آموزش، اعتبارسنجی و تست محاسبه شده‌اند.
- نمودار Loss و Accuracy به تفکیک مجموعه داده‌ها در نوت‌بوک رسم شده و تحلیل روند نشان می‌دهد که مدل به مرور دقت بالای ۸۰ %را در داده‌های اعتبارسنجی به‌دست آورده است.

۳-۲ انتخاب معماری اولیه

انتخاب معماری اولیه و بررسی تغییرات مختلف

۳-۲-۱ تغییر اندازه فیلترها

- فیلترها از اندازه 3*3 به 5*5 و 7*7 تغییر یافته‌اند.
- نتایج و تحلیل: فیلترهای بزرگ‌تر دامنه پوشش بیش‌تری دارند ولی جزئیات محلی را حذف کرده و مدل را به Underfitting در داده‌های آموزشی دچار کردند.

۳-۲-۲ تغییر Padding

- اضافه کردن Padding (Same)
- نتایج و تحلیل: اضافه کردن Padding موجب افت دقت در مجموعه تست شده است.

۳-۲-۳ تغییر Stride

- افزایش Stride از ۱ به ۲ باعث کاهش جزئیات ویژگی‌ها و کاهش دقت مدل شد اما سرعت پردازش افزایش یافت.

۳-۲-۴ تعداد لایه‌ها و فیلترها

- افزایش تعداد فیلترها به ۱۲۸ در لایه‌های اولیه باعث افزایش ظرفیت مدل شد اما به دلیل Overfitting، دقت در مجموعه تست کاهش پیدا کرد.
- کاهش فیلترها به ۱۶ باعث افت توانایی مدل در استخراج ویژگی شد.

۳-۲-۵ تغییر Pooling

- تغییر MaxPooling به AvgPooling تاثیر چشم‌گیری نداشت؛ اما AvgPooling تمایل دارد اطلاعات بیشتری حفظ کند و برای تصاویر صاف‌تر مفیدتر بود.
- افزایش اندازه MaxPooling به 3x3 باعث از دست رفتن جزئیات شد.

۳-۲-۶ توابع فعال‌سازی

- استفاده از Sigmoid به جای ReLU منجر به کندی همگرایی شد و مشکل Vanishing Gradient ظاهر شد.
- ReLU همچنان بهترین عملکرد را داشت.

۳-۲-۷ تغییر Fully Connected Layer

- افزایش نورون‌ها به ۵۱۲ باعث افزایش ظرفیت مدل و احتمال Overfitting شد.
- کاهش نورون‌ها به ۶۴ ظرفیت یادگیری را کاهش داد و مدل دقت پایین‌تری داشت.

۳-۲-۸ تحلیل و انتخاب معماری نهایی

جدول مقایسه تغییرات اعمال شده با مدل پایه

نوع تغییر	مقدار تغییر داده شده	Test Accuracy	تحلیل نتیجه
مدل پایه	-	0.9242	-
افزایش فیلتر	تغییر فیلتر به ۵	0.9445	بهتر شده
افزایش بیشتر فیلتر	تغییر فیلتر به ۷	0.9273	بهتر شده
افزودن padding	اضافه کردن padding	0.8970	بدتر شده
افزایش stride	تغییر stride به ۲	0.7853	بدتر شده
اضافه کردن لایه کانولوشنی	تغییر به سه لایه کانولوشنی	0.9300	بهتر شده
کاهش تعداد فیلترهای لایه کانولوشنی	کاهش تعداد فیلترهای لایه کانولوشنی	0.9062	بهتر شده
استفاده از AvgPooling و افزایش max pooling	افزودن به مقدار max pooling و استفاده کردن از AvgPooling	0.8107	بدتر شده
تابع فعال ساز	تغییر به sigmoid	0.6674	بدتر شده
کاهش نورون FC Layer	کاهش نورونها به ۶۴	0.9237	بدتر شده
افزایش نورون FC Layer	افزایش نورونها به ۵۱۲	0.9140	بدتر شده

- مدل پایه عملکرد بهینه و متعادل‌تری داشت و بهترین Trade-off بین دقت‌های سایر مدل‌ها را فراهم کرد.
- دقت در مدل پایه در تست ۰,۹۲۴۲ شد.
- حجم مدل نیز به این صورت شد که دو لایه کانولوشنی، یک لایه fully connected و فیلتر ۳×۳، بدون padding و stride هم ۱ است که نشان‌دهنده این است که حجم مدل پایین است و توانسته‌ایم با دو لایه کانولوشنی به دقت خوبی برسیم.
- زمان آموزش آن تقریبا ۱۲۷۵ ثانیه شد.
- زمان تست ۶ ثانیه شد
- افزایش ظرفیت مدل (فیلترها و نورون‌ها) بدون مکانیزم‌های جلوگیری از Overfitting (مثل Dropout و Regularization) منجر به Overfitting شد.
- کاهش ظرفیت مدل (فیلترها و نورون‌ها) منجر به Underfitting و ضعف در یادگیری ویژگی‌ها شد.
- تغییرات در Pooling و تابع فعال ساز نشان دادند که انتخاب ReLU و MaxPooling 2x2 همچنان انتخاب بهتری است.
- تغییر Stride و Padding به وضوح بر عملکرد مدل در آموزش و تست تاثیر گذاشتند. افزایش Padding به حفظ اطلاعات کمک کرد ولی نیاز به تنظیم دقیق داشت.

۴) آموزش شبکه عصبی

۴-۱ بهینه‌سازی مناسب

۴-۱-۱ تغییر نرخ یادگیری

نرخ یادگیری ۰,۱:

- دقت آزمون: 95.89
- زمان آموزش: 1102.25 ثانیه
- مشاهدات: بالاترین دقت آزمون در میان نرخ‌های یادگیری ثابت. دقت آموزش به سرعت بهبود یافت و در دوره ۲۰ به ۰,۹۹۳۹ رسید، اما خطای اعتبارسنجی پس از دوره ۱۰ افزایش یافت (از ۰,۲۵۲۵ به ۰,۲۶۴۱)، که نشان‌دهنده احتمال بیش‌برازش (overfitting) یا ناپایداری در دوره‌های بعدی است.

نرخ یادگیری ۰,۰۱:

- دقت آزمون: 0.9004
- زمان آموزش 1266.18 :ثانیه
- مشاهدات: دقت آزمون متوسط. دقت آموزش به طور پیوسته اما کندتر از نرخ ۰,۱ بهبود یافت و در دوره ۲۰ به ۰,۹۷۰۳ ر سید. خطای اعتبار سنجی به طور مداوم کاهش یافت، که نشان‌دهنده همگرایی پایدار اما یادگیری کند است.

نرخ یادگیری ۰,۰۰۱ :

- دقت آزمون: 0.7459
- زمان آموزش 1047.02 :ثانیه

مشاهدات:

- پایین‌ترین دقت آزمون. دقت آموزش بسیار کند بهبود یافت و تنها در دوره ۲۰ به ۰,۷۲۸۸ رسید. خطای اعتبارسنجی به تدریج کاهش یافت، اما مدل به دلیل نرخ یادگیری کوچک عملکرد ضعیفی داشت.
- نرخ یادگیری ۰,۱ با وجود اینکه مقداری بزرگ به حساب می‌آید، در این مدل عملکرد بسیار خوبی نشان داده است، احتمالاً به خاطر استفاده از SGD است که به تغییرات ناگهانی در شیب، مثل Adam واکنش نشان نمی‌دهد.

۴-۱-۲ استفاده از scheduler در تغییر نرخ یادگیری

زمان‌بندی نرخ یادگیری

- نرخ یادگیری اولیه = ۰,۱، کاهش ۰,۱ برابر هر ۵ دوره:
- دقت آزمون: ۰,۹۵۴۷
- زمان آموزش: به طور صریح ذکر نشده، اما بر اساس زمان دوره‌ها (عمدتاً حدود ۴۴ ثانیه، با برخی دوره‌های طولانی‌تر)، تقریباً ۹۰۰–۱۰۰۰ ثانیه تخمین زده می‌شود.
- مشاهدات: دقت آزمون بالا، کمی پایین‌تر از نرخ ثابت ۰,۱ اما بالاتر از نرخ‌های ۰,۰۱ و ۰,۰۰۱. زمان‌بندی با نرخ ۰,۱ شروع می‌شود، در دوره ۶ به ۰,۰۱، در دوره ۱۱ به ۰۰۱.۰ و در دوره ۱۶ به ۰,۰۰۰۱ کاهش می‌یابد. خطای اعتبارسنجی به طور پیوسته کاهش یافت و به ۰,۱۶۰۰ در دوره ۲۰ رسید، که نشان‌دهنده همگرایی پایدار و کاهش بیش‌برازش نسبت به نرخ ثابت ۰,۱ است.

مقایسه با حالت بدون Scheduler

دقت آزمون:

- نرخ ثابت ۰,۱ :بالاترین دقت آزمون (۰,۹۵۸۹)، کمی بهتر از زمان‌بندی (۰,۹۵۴۷). با این حال، خطای اعتبارسنجی بالا در دوره‌های بعدی نشان‌دهنده احتمال بیش‌برازش است.

- زمان‌بندی :دقت آزمون (۰,۹۵۴۷) بسیار نزدیک به نرخ ثابت ۰,۱ و به طور قابل‌توجهی بهتر از نرخ‌های ثابت ۰,۰۱ (۰,۹۰۰۴) و ۰,۰۰۱ (۰,۷۴۵۹). زمان‌بندی با تعادل بین یادگیری سریع اولیه (با نرخ بالا) و تنظیم دقیق (با نرخ پایین)، تعمیم‌پذیری خوبی ارائه می‌دهد.
- نرخ‌های ثابت ۰,۰۱ و ۰,۰۰۱ :دقت آزمون بسیار پایین‌تر، که نشان می‌دهد این نرخ‌های یادگیری برای آموزش مؤثر مدل در ۲۰ دوره خیلی کوچک بودند.

زمان آموزش:

- نرخ ثابت ۰,۱ 1102.25 :ثانیه.
- نرخ ثابت ۰,۰۱ 1266.18 :ثانیه (طولانی‌ترین به دلیل همگرایی کندتر).
- نرخ ثابت ۰,۰۰۱ 1047.02 : ثانیه (کوتاه‌ترین در میان نرخ های ثابت، احتمالاً به دلیل به‌روزرسانی‌های ساده‌تر با نرخ کوچک).
- زمان‌بندی: تخمین زده شده در حدود ۹۰۰–۱۰۰۰ ثانیه (بر اساس زمان دوره‌ها). زمان آموزش روش زمان‌بندی کوتاه‌تر از نرخ‌های ثابت ۰,۱ و ۰,۰۱ است، زیرا با تنظیم نرخ یادگیری سریع‌تر همگرا می‌شود.

نتیجه گیری نهایی

زمان‌بندی نرخ یادگیری از نظر پایداری، تعمیم پذیری و احتمالاً کارایی آموزش نسبت به نرخ‌های یادگیری ثابت عملکرد بهتری دارد و دقتی در آزمون (۰,۹۵۴۷) نزدیک به بهترین نرخ یادگیری ثابت (۰,۹۵۸۹، نرخ ۰,۱) با خطای اعتبارسنجی کمتر (۰,۱۶۰۰) به دست می‌آورد. توانایی زمان‌بندی در تنظیم پویا نرخ یادگیری، آن را به رویکرد ترجیحی برای این کار تبدیل می‌کند، زیرا یادگیری سریع نرخ‌های بالا را با توانایی تنظیم دقیق نرخ‌های پایین ترکیب می‌کند و منجر به آموزش قوی و کارآمد می‌شود.

۴-۱-۳ امتحان کردن ضرایب مختلف رگولاسیون

بر اساس خروجی کد ارائه‌شده، مدل با ضرایب رگولاسیون L2 در بازه [۰, ۱] با گام ۱.۰ آموزش داده شده است. دقت‌های آزمون برای هر مقدار L2 و مقدار بهینه به شرح زیر است:

- L2 = 0.0: دقت آزمون = ۰,۹۱۲۵

- L2 = 0.1: دقت آزمون = ۰,۶۵۲۷

- L2 = 0.2: دقت آزمون = ۰,۵۹۵۳

- L2 = 0.3: دقت آزمون = ۰,۵۹۵۳

- L2 = 0.4: دقت آزمون = ۰,۵۹۵۳

- L2 = 0.5: دقت آزمون = ۰,۵۹۵۳

- L2 = 0.6: دقت آزمون = ۰,۵۹۵۳

- L2 = 0.7: دقت آزمون = ۰,۵۹۵۳

- L2 = 0.8: دقت آزمون = ۰,۵۹۵۳

- L2 = 0.9: دقت آزمون = ۰,۵۹۵۳

- L2 = 1.0: دقت آزمون = ۰,۵۹۵۳

مقدار بهینه: L2: 0.0

دقت آزمون بهینه: ۰,۹۱۲۵

نتیجه گیری نهایی:

مقدار بهینه L2 برابر با ۰,۰با دقت آزمون ۰,۹۱۲۵ است. مقادیر بالاتر L2 دقت را به طور قابل‌توجهی کاهش داده‌اند، که نشان می‌دهد رگولاسیون L2 در این مورد ضروری نیست.

۴-۱-۴ تغییر و مقایسه با سایر تابع‌های بهینه‌ساز

با توجه به کد و خروجی ارائه‌شده، مدل با استفاده از مقدار بهینه رگولاسیون L2 (یعنی L2=0.0) و دو بهینه‌ساز مختلف (SGD و Adam) با نرخ یادگیری اولیه ۰,۱ و زمان‌بندی نرخ یادگیری (کاهش ۰,۱ برابر هر ۵ دوره) آموزش داده شده است.

در ادامه، نتایج این دو بهینه‌ساز را تحلیل و مقایسه می‌کنم.

SGD:

- دقت آزمون: ۰,۹۵۷۵
- مشـــاهدات: SGD با نرخ یادگیری اولیه ۰,۱ و زمان‌بندی نرخ یادگیری عملکرد بســـیار خوبی داشـــته و به دقتی نزدیک به بهینه (مشـــابه دقت‌های قبلی با نرخ یادگیری ۰,۱ و زمان‌بندی) رسـیده اسـت. این نشـان می‌دهد که ترکیب SGD با زمان‌بندی نرخ یادگیری برای این مدل و مجموعه داده بسیار مؤثر است.

Adam:

- دقت آزمون: ۰,۵۹۵۳
- مشاهدات: Adam عملکرد بسیار ضعیفی داشته و دقتی مشابه حالتی که رگولاسیون L2 بیش از حد قوی بود (مثل $L2 \geq 0.2$ در آزمایش‌های قبلی) ارائه داده است. این دقت بسیار پایین‌تر از SGD و حتی پایین‌تر از دقت بدون رگولاسیون با SGD (۰,۹۱۲۵ در آزمایش‌های قبلی) است.

نتیجه گیری نهایی:

- نرخ یادگیری ۰,۱ و زمان‌بندی برای SGD مناســـب اســـت، اما برای Adam نامناســـب بوده و احتمالاً باعث ناپایداری یا همگرایی به یک راه‌حل ضعیف شده است.
- Adam معمولاً نیازی به زمان‌بندی نرخ یادگیری ندارد، زیرا مکانیزم‌های داخلی آن (مومنتوم و RMSprop) نرخ یادگیری مؤثر را به طور خودکار تنظیم می‌کنند.
- با $L2=0.0$، ممکن است مدل کمی مستعد بیش‌برازش باشد، اما دقت بالای آزمون (۰.۹۵۷۵) و نتایج مشـــابه در آزمایش‌های قبلی (۰,۹۵۴۷ با زمان‌بندی) نشـــان می‌دهد که زمان‌بندی نرخ یادگیری به خوبی از بیش‌برازش جلوگیری کرده است.
- کاهش تدریجی نرخ یادگیری به مدل اجازه داده تا در مراحل پایانی آموزش تنظیم دقیق انجام دهد و تعمیم‌پذیری خوبی داشته باشد.
- برای این مدل و مجموعه داده، اســـتفاده از SGD با نرخ یادگیری اولیه ۰,۱ و زمان‌بندی نرخ یادگیری توصـــیه می‌شـــود، زیرا دقت بالا و پایداری خوبی ارائه می‌دهد. اگر بخواهیم Adam را

آزمایش کنیم، باید نرخ یادگیری را به ۰,۰۰۱ یا ۰,۰۰۰۱ کاهش دهیم و زمان‌بندی نرخ یادگیری را حذف کنیم تا شانس دستیابی به دقت بالاتر فراهم شود.

۴-۲ لایه‌های کمکی

۴-۲-۱ اضافه کردن لایه BatchNorm

برای بررسی تأثیر افزودن لایه‌های BatchNormalization به معماری شبکه عصبی، نتایج خروجی کد ارائه‌شده (که شامل BatchNormalization است) را با نتایج قبلی (بدون BatchNormalization) مقایسه می‌کنم. سپس تأثیر این لایه‌ها را بر عملکرد مدل، شامل دقت آزمون، سرعت همگرایی، پایداری و تعمیم‌پذیری، تحلیل می‌کنم.

بر اساس خروجی کد ارائه‌شده:

- دقت آزمون: ۰,۹۵۸۲
- خطای اعتبارسنجی (دوره ۲۰): ۰,۱۶۴۱

مقایسه با نتایج قبلی (بدون BatchNormalization)

- دقت آزمون: ۰,۹۵۸۹
- خطای اعتبارسنجی (دوره ۲۰): ۰,۲۶۴۱

توصیه: استفاده از BatchNormalization برای این مدل و مجموعه داده به شدت توصیه می‌شود، زیرا دقت بالا، تعمیم‌پذیری خوب و پایداری را با حداقل نیاز به تنظیم hyperparameter ها فراهم می‌کند. برای بهبود بیشتر، می‌توان اندازه دسته را بهینه کرد یا BatchNormalization را با بهینه‌سازهای دیگر (مثل Adam با نرخ یادگیری مناسب) آزمایش کرد.

۴-۲-۲ اضافه کردن لایه Dropout

برای بررسی تأثیر افزودن لایه Dropout با نرخ‌های ۰,۲ و ۰,۵ به معماری شبکه عصبی، نتایج خروجی کدهای ارائه‌شده (که شامل Dropout است) را با نتایج قبلی (بدون Dropout) و همچنین با یکدیگر

مقایسه می‌کنم. سپس تأثیر این لایه‌ها را بر عملکرد مدل، شامل دقت آزمون، سرعت همگرایی، پایداری، تعمیم‌پذیری و بیش‌برازش، تحلیل می‌کنم.

نتایج با Dropout

۱. Dropout با نرخ ۰,۲

- دقت آزمون: ۰,۹۱۰۶
- دقت اعتبارسنجی (دوره ۲۰): ۰,۹۱۱۰
- خطای اعتبارسنجی (دوره ۲۰): ۰,۲۲۴۸

مشاهدات

دقت آموزش از ۰,۶۰۰۸ در دوره ۱ به ۰,۹۳۳۰ در دوره ۲۰ افزایش یافته است، که نشان‌دهنده یادگیری مناسب است، اما به اندازه مدل بدون Dropout (یا با BatchNormalization) بالا نیست.

دقت اعتبارسنجی به طور پیوسته بهبود یافته و در دوره ۲۰ به ۰,۹۱۱۰ رسیده است، که با دقت آزمون (۰,۹۱۰۶) بسیار نزدیک است و نشان‌دهنده تعمیم‌پذیری خوب است.

خطای اعتبارسنجی به ۰,۲۲۴۸ کاهش یافته، اما در مقایسه با مدل‌های قبلی (مثل BatchNormalization با خطای ۰,۱۶۴۱) کمی بالاتر است.

همگرایی نسبتاً کندتر از مدل با BatchNormalization است، اما پایداری خوبی دارد.

۲. Dropout با نرخ ۰,۵

- دقت آزمون: ۰,۸۸۵۸
- دقت اعتبارسنجی (دوره ۲۰): ۰,۸۸۷۳
- خطای اعتبارسنجی (دوره ۲۰): ۰,۲۶۸۵

مشاهدات:

- دقت آموزش از ۰,۶۰۱۲ در دوره ۱ به ۰,۸۷۴۶ در دوره ۲۰ افزایش یافته، که پایین‌تر از مدل با Dropout 0.2 است.
- دقت اعتبارسنجی به ۰,۸۸۷۳ رسیده، که با دقت آزمون (۰,۸۸۵۸) نزدیک است و نشان‌دهنده تعمیم‌پذیری مناسب است، اما دقت کلی پایین‌تر از Dropout 0.2 است.
- خطای اعتبارسنجی به ۰,۲۶۸۵ کاهش یافته، اما بالاتر از Dropout 0.2 (۰,۲۲۴۸) و مدل‌های قبلی (مثل BatchNormalization با ۰,۱۶۴۱) است.
- همگرایی کندتر از Dropout 0.2 است و به دلیل نرخ بالای Dropout، مدل کمتر یاد گرفته است.
- مقایسه با نتایج قبلی (بدون Dropout)
- تأثیر:
- در Dropout 0.2، حذف ۲۰٪ از نورون‌ها در آموزش باعث شده مدل به الگوهای قوی‌تر و عمومی‌تر وابسته شود، که به دقت آزمون خوب (۰,۹۱۰۶) منجر شده است.
- در Dropout 0.5، حذف ۵۰٪ از نورون‌ها در آموزش یادگیری را بیش از حد محدود کرده و مدل نتوانسته الگوهای کافی را یاد بگیرد، که به دقت پایین‌تر (۰,۸۸۵۸) منجر شده است.
- از آنجا که Dropout در پیش‌بینی حذف می‌شود، تأثیر آن در این مرحله غیرمستقیم است و به کیفیت وزن‌های آموزش‌دیده بستگی دارد.

۴-۲-۳ بررسی تاثیر همزمان دو لایه Batchnorm و Dropout

برای بررسی تأثیر همزمان افزودن لایه‌های BatchNormalization و Dropout (با نرخ ۰,۲) به معماری شبکه عصبی، نتایج خروجی کد ارائه‌شده را با نتایج قبلی (مدل با BatchNormalization تنها، Dropout تنها، و بدون این لایه‌ها) مقایسه می‌کنم. سپس تأثیر این ترکیب بر دقت آزمون، سرعت همگرایی، پایداری، تعمیم‌پذیری، بیش‌برازش و نقش آن در مرحله پیش‌بینی را تحلیل می‌کنم.

نتایج با BatchNormalization و Dropout (نرخ 0.2)

بر اساس خروجی کد:

- دقت آزمون: 0.7915

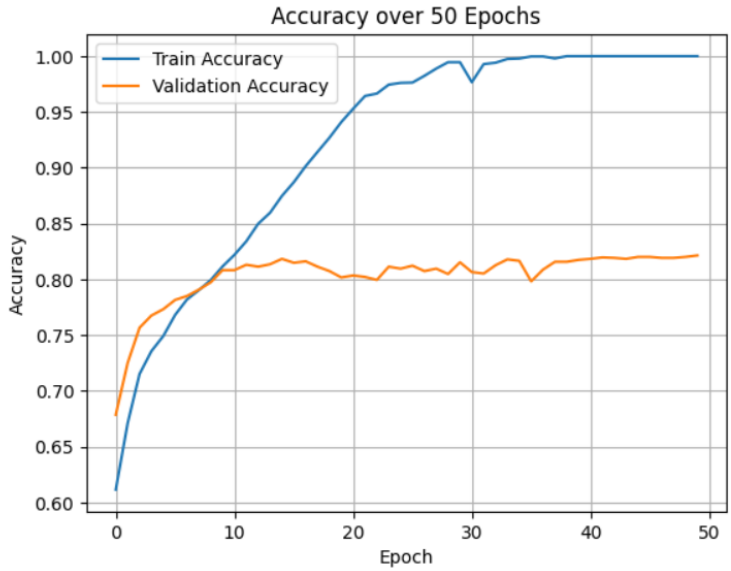
- دقت اعتبارسنجی (دوره ۲۰): 0.8066
- خطای اعتبارسنجی (دوره ۲۰): 0.6290

مشاهدات :

- دقت آموزش از ۰,۶۶۵۸ در دوره ۱ به ۰,۹۹۱۱ در دوره ۲۰ افزایش یافته است، که نشان‌دهنده یادگیری قوی در داده‌های آموزشی است.
- دقت اعتبارسنجی در دوره‌های اولیه افزایش یافته، اما در دوره ۲۰ به ۰,۸۰۶۶ رسیده و نوسانات زیادی داشته (مثلاً کاهش به ۰,۵۹۹۱ در دوره ۳ و افزایش به ۰,۸۲۷۱ در دوره ۱۷).
- خطای اعتبار سنجی در دوره ۲۰ به ۰,۶۲۹۰ رسیده، که بسیار بالاتر از مدل‌های قبلی است و نشان‌دهنده تعمیم‌پذیری ضعیف است.
- دقت آزمون (۰,۷۹۱۵) پایین‌تر از دقت آموزش (۰,۹۹۱۱) و حتی اعتبارسنجی (۰,۸۰۶۶) است، که نشان‌دهنده مشکل در تعمیم‌پذیری و احتمال بیش‌برازش یا ناپایداری است.
- ترکیب BatchNormalization و Dropout 0.2 پایداری مدل را کاهش داده و تعمیم‌پذیری ضعیفی ارائه کرده است.
- این نتیجه ممکن است به دلیل تداخل بین BatchNormalization (که فعال‌سازی‌ها را نرمال‌سازی می‌کند) و Dropout (که نورون‌ها را به طور تصادفی حذف می‌کند) باشد، که می‌تواند گرادیان‌ها را ناپایدار کند .

۴-۳ توقف مناسب آموزش

۴-۳-۱ تغییر epoch



نتایج با ۵۰ دوره

بر اساس خروجی کد:

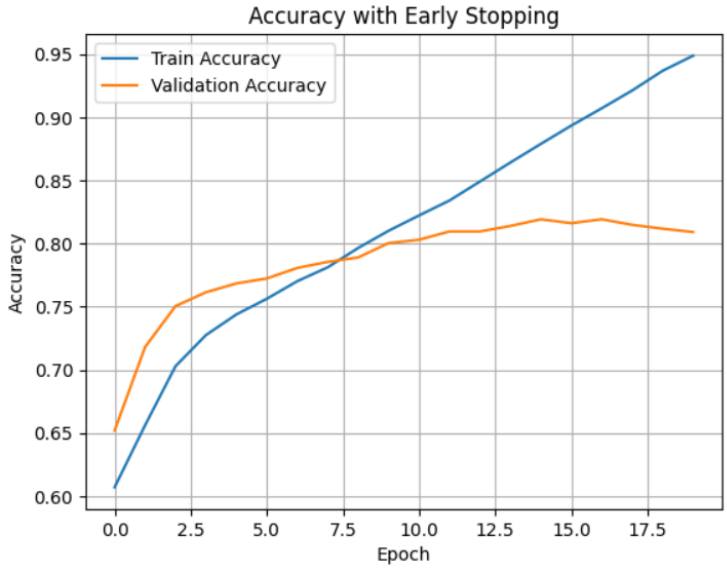
- **دقت آزمون**: 0.8112
- **دقت اعتبارسنجی (دوره ۵۰)**: 0.8214
- **خطای اعتبارسنجی (دوره ۵۰)**: 1.0779

مشاهدات:

- دقت آموزش از ۰٫۶۰۱۱ در دوره ۱ به ۱٫۰۰۰۰ در دوره ۳۹ و تا دوره ۵۰ ثابت مانده است، که نشان‌دهنده یادگیری کامل داده‌های آموزشی است.
- دقت اعتبارسنجی به طور کلی افزایش یافته، اما پس از دوره ۱۵ (دقت ۰٫۸۱۸۴) به حالت تقریباً ثابت (حدود ۰٫۸۲) رسیده و در دوره ۵۰ به ۰٫۸۲۱۴ رسیده است.

- خطای اعتبارسنجی پس از کاهش اولیه (به ۰,۳۹۵۹ در دوره ۱۴) به طور پیوسته افزایش یافته و در دوره ۵۰ به ۱,۰۷۷۹ رسیده است.
- دقت آزمون (۰,۸۱۱۲) و دقت اعتبارسنجی (۰,۸۲۱۴) نزدیک هستند، اما بسیار پایین‌تر از دقت آموزش (۱,۰۰۰۰) هستند، که نشان‌دهنده بیش‌برازش قابل‌توجه است.

۴-۳-۲ استفاده از Early Stopping



نتایج باEarly Stopping بر اساس خروجی کد:

- **دقت آزمون**: 0.8029
- **دقت اعتبارسنجی (آخرین دوره)**: 0.8092
- **خطای اعتبارسنجی (آخرین دوره)**: 0.4892
- **تعداد دوره‌های اجرا شده:** ۲۰ دوره (آموزش به دلیل Early Stopping در دوره ۲۰ متوقف شده، زیرا خطای اعتبارسنجی پس از دوره ۱۵ (۰,۴۰۵۴) به مدت ۵ دوره بهبود نیافته است).

مشاهدات :

- دقت آموزش از ۰,۶۰۱۷ در دوره ۱ به ۰,۹۴۴۱ در دوره ۲۰ افزایش یافته است.

- دقت اعتبارســـنجی به ۰,۸۱۹۳ در دوره ۱۵ رســـیده و ســـپس کاهش یافته و در دوره ۲۰ به ۰,۸۰۹۲ رسیده است.
- خطای اعتبارســـنجی پس از کاهش به ۰,۴۰۵۴ در دوره ۱۵، افزایش یافته و در دوره ۲۰ به ۰,۴۸۹۲ رسیده است.
- Early Stopping با patience=5 و restore_best_weights=True وزن های بهترین مدل (احتمالاً دوره ۱۵ با خطای ۰,۴۰۵۴) را بازگرداند.

۴-۳-۳ استفاده از ModelCheckpoint (ذخیره بهترین مدل بر اساس دقت اعتبارسنجی)

مدل با SGD (نرخ یادگیری ۰,۰۱)، بدون رگولاســـیون، با ModelCheckpoint (نظارت بر val_accuracy, save_best_only=True) برای ۵۰ دوره آموزش داده شد.

نتایج :

- دقت آزمون: ۰,۸۱۱۲
- دقت اعتبارسنجی (بهترین مدل): ۰,۸۱۴۹ (۵۰ دوره)
- خطای اعتبارسنجی (بهترین مدل): ۱,۰۸۵۲ (۵۰ دوره)
- دقت آموزش (دوره ۵۰): ۱,۰۰۰۰
- تعداد دوره‌ها:(50 (بهترین مدل در دوره ۵۰ ذخیره شد)

مشاهدات:

- دقت اعتبارسنجی از ۰,۶۳۵۹ (دوره ۱) به ۰,۸۱۴۹ (دوره ۵۰) افزایش یافت، اما پس از دوره ۱۴ (۰,۸۱۲۳) نوسانات داشت.
- خطای اعتبارسنجی از ۰,۴۱۲۰ (دوره ۱۳) به ۱,۰۸۵۲ افزایش یافت، که نشان‌دهنده بیش‌برازش است.
- هزینه محاسباتی بالا (~۱۸۵۰–۲۱۵۰ ثانیه).

- ModelCheckpoint بهترین مدل را بر اســاس دقت اعتبارســنجی (۰,۸۱۴۹) ذخیره کرد، اما دقت آزمون (۰,۸۱۱۲) مشابه ۵۰ دوره بدون ModelCheckpoint بود.

۴-۳-۴ مقایسه عملکرد سه روش

- **50 دوره بدون رگولاسیون** :بدترین عملکرد از نظر بیش‌برازش و تعمیم‌پذیری (خطای اعتبارسنجی ۱,۰۷۷۹)، با هزینه محاسباتی بالا و بدون مزیت نسبت به توقف زودهنگام.
- **Early Stopping**: بهترین کنترل بر بیش‌برازش (خطای اعتبارسنجی ~۰,۴۰۵۴)، اما دقت آزمون پایین (۰,۸۰۲۹) به دلیل توقف زودهنگام. با تنظیم patience (مثل ۱۰)، می‌توانست بهتر عمل کند .
- **ModelCheckpoint**: دقت اعتبارسنجی کمی بالاتر (۰,۸۱۴۹)، اما بیش‌برازش شدید (خطای اعتبارسنجی ۱,۰۸۵۲) و هزینه محاسباتی بالا، مشابه ۵۰ دوره بدون رگولاسیون .

مقایسه کلی سه بخش			
معیار	50 اپوک (بدون رگولاسیون)	Early Stopping (توقف در اپوک 20)	ModelCheckpoint (50 اپوک)
دقت آزمون	0.8112	0.8029	0.8112
دقت اعتبارسنجی	0.8214 (اپوک 50)	0.8193 (اپوک 15, بهترین)	0.8149 (اپوک 50, بهترین)
خطای اعتبارسنجی	1.0779 (اپوک 50)	~0.4054 (اپوک 15, بهترین)	1.0852 (اپوک 50)
دقت آموزش	1.0000 (اپوک 50)	0.9441 (اپوک 20)	1.0000 (اپوک 50)
بیش‌برازش	شدید	متوسط	شدید
تعمیم‌پذیری	ضعیف	متوسط	ضعیف
پایداری	پایین (نوسانات زیاد)	متوسط (توقف زودهنگام)	پایین (نوسانات زیاد)
هزینه محاسباتی	بالا (~1850–2200 ثانیه)	پایین (~740–820 ثانیه)	بالا (~1850–2150 ثانیه)
بهترین اپوک	~15–20 (بر اساس نمودار)	15 (بر اساس val_loss)	50 (بر اساس val_accuracy)

نتیجه نهایی: هترین زمان توقف آموزش در این مدل بدون رگولاسیون حدود **دوره ۱۴–۱۵** است، جایی که خطای اعتبارسنجی کمینه و دقت اعتبارسنجی بالاست. برای عملکرد بهتر، ترکیب Early Stopping، ModelCheckpoint و رگولاسیون (مثل BatchNormalization) توصیه می‌شود، که می‌تواند دقت آزمون را به سطح مدل‌های رگولاریزه‌شده (تا ۰,۹۵۸۲) نزدیک کند.

۴-۴ تنظیم مدل

نتایج Keras Tuner

استفاده از Keras Tuner (RandomSearch) برای بهینه‌سازی نرخ یادگیری (lr) و ضریب رگولاسیون L2 (l2) با Early Stopping (monitor='val_accuracy', patience=3, restore_best_weights=True).

فضای جست‌وجو :

- نرخ یادگیری: [1e-4, 1e-1] (مقیاس لگاریتمی)
- ضریب L2: [1e-5, 1e-2] (مقیاس لگاریتمی)

جزئیات جست‌وجو :

- تعداد آزمایش‌ها: ۱۰ (max_trials=10)
- دوره‌های هر آزمایش: حداکثر ۱۰ با Early Stopping
- بهترین دقت اعتبارسنجی: ۰,۸۰۸۶
- بهترین هایپرپارامترها: lr=0.0914, l2=0.000169

آموزش نهایی :

- مدل با بهترین هایپرپارامترها برای ۲۰ دوره آموزش داده شد.
- Early Stopping در دوره ۷ (پس از دوره ۴ با دقت اعتبارسنجی ۰,۸۰۰۳) توقف کرد.

نتایج نهایی :

- دقت آزمون: ۰,۸۰۴۷
- دقت اعتبارسنجی (بهترین): ۰,۸۰۰۳ (دوره ۴)
- خطای اعتبارسنجی (بهترین): ۰,۴۳۶۷ (دوره ۴)
- دقت آموزش (آخرین دوره): ۰,۹۳۸۳

- خطای آموزش (آخرین دوره): ۰,۱۸۲۷
- تعداد دوره‌ها: 7 (به دلیل Early Stopping)

هزینه محاسباتی :

- جست‌وجوی هایپرپارامترها: ~۵۵ دقیقه و ۳۳ ثانیه (~۳۳۳۳ ثانیه).
- آموزش نهایی: ~۲۳۷ ثانیه (۷ دوره × ~۳۴ ثانیه/دوره).
- مجموع: ~۳۵۷۰ ثانیه.

مشاهدات:

- دقت آموزش از ۰,۶۱۳۶ (دوره ۱) به ۰,۹۳۸۳ (دوره ۷) افزایش یافت.
- دقت اعتبارسنجی در دوره ۴ به ۰,۸۰۰۳ رسید، اما پس از آن کاهش یافت (۰,۷۹۶۳ در دوره ۷).
- خطای اعتبارسنجی از ۰,۴۳۶۷ (دوره ۴) به ۰,۶۵۵۳ (دوره ۷) افزایش یافت، که نشان‌دهنده شروع بیش‌برازش است.
- Early Stopping با patience=3 آموزش را در دوره ۷ متوقف کرد و وزن های دوره ۴ (بهترین دقت اعتبارسنجی) را بازگرداند.
- دقت آزمون (۰,۸۰۴۷) و دقت اعتبارسنجی (۰,۸۰۰۳) نزدیک هستند، اما پایین‌تر از دقت آموزش (۰,۹۳۸۳)، که نشان‌دهنده بیش‌برازش متوسط است.

تحلیل نتایج Keras Tuner نسبت به بخش‌های قبلی

مقایسه با Keras Tuner				
معیار	50 اپوک	Early Stopping	ModelCheckpoint	Keras Tuner
دقت آزمون	0.8112	0.8029	0.8112	0.8047
دقت اعتبارسنجی	0.8214	0.8193	0.8149	0.8003
خطای اعتبارسنجی	1.0779	0.4054~	1.0852	0.4367
دقت آموزش	1.0000	0.9441	1.0000	0.9383
بیش‌برازش	شدید	متوسط	شدید	متوسط
تعمیم‌پذیری	ضعیف	متوسط	ضعیف	متوسط
پایداری	پایین	متوسط	پایین	متوسط
هزینه محاسباتی	~1850–2200 ثانیه	~740–820 ثانیه	~1850–2150 ثانیه	~3570 ثانیه
بهترین اپوک	14–15	15	13–14	4

- **دقت آزمون** Keras Tuner (0.8047) :مشابه Early Stopping (0.8029) و کمی پایین‌تر از ۵۰ دوره و ModelCheckpoint (0.8112) بود، اما بسیار پایین‌تر از مدل‌های رگولاریزه‌شده (تا ۰,۹۵۸۲).
- **بیش‌برازش** Keras Tuner :با L2 و Early Stopping بیش‌برازش را نسبت به ۵۰ دوره و ModelCheckpoint کاهش داد، اما همچنان بیش‌برازش متوسط داشت.
- **تعمیم‌پذیری** :خطای اعتبارسنجی (۰,۴۳۶۷) بهتر از ۵۰ دوره و ModelCheckpoint (1.0779–1.0852) بود، اما ضعیف‌تر از Early Stopping (~0.4054) و مدل‌های رگولاریزه‌شده.(0.1641)
- **پایداری** :پایداری متوسط، مشابهEarly Stopping ، اما نرخ یادگیری بالا باعث نوسانات شد.
- **هزینه محاسباتی** :هزینه بالا (~۳۵۷۰ ثانیه) بدون توجیه با توجه به دقت پایین.

۴-۵ انتخاب نهایی مدل

با توجه به موارد قبلی که نشان دادیم تحلیل می‌کنیم و معماری نهایی خود را ارائه می‌دهیم:

با اضافه شدن dropout 0.2 و batch میزان دقت تست پایین آمد. اگر فقط dropout 0.5 را اعمال می‌کردیم اوضاع بهتر میشد ولی باز هم مدل اولیه بهتر بود. اگر فقط dropout 0.2 را اعمال می‌کردیم نتیجه تست ۰,۹۱۰۵ میشد که نتیجه بسیار خوبی است و تقریبا در حد مدل پایه است. یه روش دیگر که اعمال کردیم اضافه کردن batch normalization layer بصورت تغییر دیگری بود که دقت در داده‌های تست به ۰,۹۵۸۲ رسید. بهینه‌ساز را هم با توجه به نتایجی که بدست آمد SGD بهترین گزینه برای ما است. L2 regularization نیز اعمال کردیم و دقت ۰,۹۱۲۵ را بدست آوردیم. استفاده از learning rate scheduler نیز کمک‌کننده بود و دقت ۰,۹۵۴۶ را به ما داد. اما وقتی نیز از learning rate با مقدار ثابت ۰,۱ استفاده کردیم نتیجه تقریبا مشابهی گرفتیم که با توجه به اینکه اگر مقدار learning rate ثابت باشد محاسباتمان کم می‌شود بهتر است از learning rate ثابت ۰,۱ استفاده کنیم. استفاده از early stopping نیز نتیجه بهتری به ما نداد. نتیجه کلی بصورت زیر است:

- دو لایه کانولوشنی با تنظیمات مدل پایه
- استفاده از batch normalization layer
- استفاده از بهینه ساز SGD
- استفاده از learning rate ثابت ۰,۱
- می‌توان از L2 regularizatoin نیز استفاده کرد.

۴-۶ انتقال یادگیری

مدل MobileNetV2 با وزن‌های پیش‌آموزش‌دیده ImageNet، لایه‌های پایه فریزشده (trainable=False)، و افزودن لا یه های GlobalAveragePooling2D، Dense(128, relu)، و Dense(2, softmax)برای طبقه‌بندی دو کلاس (خوشحال/ناراحت).

- **بهینه‌ساز** SGD :با نرخ یادگیری ۰,۰۱.
- **آموزش** 20 :دوره بدون Early Stopping یا رگولاسیون اضافی.

نتایج :

- **دقت** آزمون: ۰,۶۸۷۸
- دقت اعتبارسنجی (آخرین دوره): ۰,۶۸۸۱
- خطای اعتبارسنجی (آخرین دوره): ۰,۹۰۲۵
- دقت آموزش (آخرین دوره): ۰,۹۹۰۷
- خطای آموزش (آخرین دوره): ۰,۰۸۳۲
- بهترین دقت اعتبارسنجی: ۰,۶۸۸۶ (دوره ۱۵)
- بهترین **خطای اعتبارسنجی**: ۰,۶۲۳۶ (دوره ۳)

مشاهدات :

- دقت آموزش از ۰,۶۳۴۳ (دوره ۱) به ۰,۹۹۰۷ (دوره ۲۰) افزایش یافت.
- دقت اعتبارسنجی از ۰,۶۴۱۷ (دوره ۱) به ۰,۶۸۸۶ (دوره ۱۵) رسید، اما در دوره ۲۰ به ۰,۶۸۸۱ کاهش یافت.
- خطای اعتبارسنجی از ۰,۶۲۳۶ (دوره ۳) به ۰,۹۰۲۵ (دوره ۲۰) افزایش یافت.
- هشدار: اندازه ورودی (img_height, img_width) با اندازه پیش‌فرض MobileNetV2 (224x224)مطابقت نداشته، اما وزن‌های ۲۲۴ 224xبارگذاری شدند.

نتیجه گیری:

مدل MobileNetV2 با دقت آزمون ۰,۶۸۷۸ عملکرد ضعیفی داشت، عمدتاً به دلیل بیش‌برازش شدید، عدم پیش‌پردازش مناسب، و عدم رگولاسیون. دوره ۵–۷ بهترین نقطه برای توقف بود، و با اعمال پیش‌پردازش، رگولاسیون، و Early Stopping می‌توان عملکرد را بهبود داد.

۴-۷ تاثیر پایگاه داده بر آموزش

گزارش و تحلیل نتایج

- دقت آزمون : ۰,۷۸۰۲ (اندازه دسته‌های داده‌های آزمون تقریباً یکسان).
- دقت اعتبارسنجی (آخرین دوره) : ۰,۷۹۲۸
- خطای اعتبارسنجی (آخرین دوره) : ۰,۵۵۱۸
- دقت آموزش (آخرین دوره) : ۰,۹۵۳۳
- بهترین دقت اعتبارسنجی : ۰,۷۹۷۲ (دوره ۱۸)
- بهترین خطای اعتبارسنجی : ۰,۴۴۶۴ (دوره ۱۳)
- هزینه محاسباتی : ~۷۶۶ ثانیه (~۱۲ دقیقه و ۴۶ ثانیه، میانگین ۳۸ ثانیه/دوره).

نتیجه‌گیری:

دقت آزمون ۰,۷۸۰۲ نشان‌دهنده عملکرد متوسط با بیش‌برازش قابل‌توجه است. وزن‌های کلاسی به مدیریت داده‌های نامتقارن کمک کردند، اما عدم رگولاسیون و آموزش طولانی تعمیم‌پذیری را کاهش داد. دوره ۱۲–۱۳ بهترین نقطه برای توقف بود. با افزودن رگولا سیون و Early Stopping می‌توان دقت و تعمیم‌پذیری را بهبود داد.

۵) استفاده از مدل شبکه عصبی

۵-۱ دقت بر روی سایر پایگاه داده

نتایج روی affectnet

- دقت آزمون (داده‌های اولیه) : ۰,۷۷۵۰
- دقت آزمون (داده‌های affectnet) : ۰,۶۵۵۰
- دقت اعتبارسنجی (آخرین دوره) : ۰,۷۹۳۳
- خطای اعتبارسنجی (آخرین دوره) : ۰,۵۱۸۵

- دقت آموزش (آخرین دوره) : ۰,۹۴۳۳
- بهترین دقت اعتبارسنجی : ۰,۸۰۸۶ (دوره ۱۶)
- بهترین خطای اعتبارسنجی : ۰,۴۳۰۴ (دوره ۱۴)
- هزینه محاسباتی : ~۷۶۹ ثانیه (~۱۲ دقیقه و ۴۹ ثانیه، میانگین ~۳۸,۵ ثانیه/دوره).

نتیجه‌گیری:

دقت آزمون ۰,۷۷۵۰ (داده‌های اولیه) و ۰,۶۵۵۰ (affectnet) نشان‌دهنده عملکرد متوسط با بیش‌برازش و تعمیم‌پذیری ضعیف است. دوره ۱۴–۱۶ بهینه بود. افزودن رگولاسیون و Early Stopping می‌تواند دقت و تعمیم‌پذیری را بهبود دهد.

نتایج روی JAFFE

- دقت آزمون (داده‌های اولیه) : ۰,۷۸۶۸
- دقت آزمون (داده‌های JAFFE) : ۰,۷۲۵۸
- دقت اعتبارسنجی (آخرین دوره) : ۰,۷۹۷۶
- خطای اعتبارسنجی (آخرین دوره) : ۰,۵۷۲۹
- دقت آموزش (آخرین دوره) : ۰,۹۵۱۶
- بهترین دقت اعتبارسنجی : ۰,۸۰۲۰ (دوره ۱۹)
- بهترین خطای اعتبارسنجی : ۰,۴۴۸۲ (دوره ۱۱)
- هزینه محاسباتی : ~۷۶۹ ثانیه (~۱۲ دقیقه و ۴۹ ثانیه، میانگین ~۳۸,۵ ثانیه/دوره).

نتیجه‌گیری:

دقت آزمون ۰,۷۸۶۸ (داده‌های اولیه) و ۰,۷۲۵۸ (JAFFE) نشان‌دهنده عملکرد متوسط با بیش‌برازش و تعمیم‌پذیری محدود است. دوره ۱۱–۱۴ بهینه بود. رگولاسیون و Early Stopping می‌توانند عملکرد را بهبود دهند.

Abstract

This exercise focused on the design and training of convolutional neural networks (CNNs). Initially, we analyzed four datasets, examining aspects such as image types, data distribution across categories, labeling methods, and the presence of corrupted data. Based on this analysis, we selected two compatible datasets for combination.

We then performed preprocessing steps: resizing images to uniform dimensions, renaming files for consistency, and organizing the data into a structured directory. The dataset was split into training, validation, and test sets.

A CNN model was constructed and trained, followed by performance evaluation. Subsequently, we experimented with various architectural modifications, including adjustments to filters, padding, stride, the addition of layers, changes in pooling strategies (e.g., increasing max pooling size and incorporating average pooling), and alterations to activation functions and fully connected layers.

We also explored different optimizers and assessed the impact of each change on model performance. The results were analyzed in detail, with visual aids such as charts and tables employed to illustrate outcomes and facilitate comparisons.

The code and detailed report are available at:

https://colab.research.google.com/drive/1exZxLRBveWJA1QPUpf55X3r0d6XD4iF5?usp=sharing

Key Words: data preprocessing, deep learning, emotion recognition, convolutional neural networks, image processing

Amirkabir University of Technology
(Tehran Polytechnic)

Department of Electrical Engineering

Homework 1

Convolutional Neural Networks

By
Mohammadreza Shahrestani
Hossein Khamooshi

Supervisor
Dr. Saeed Sharifian

April 2025