محمد جواد رنجبر

۸۱۰۱۰۱۱۷۳

تمرین پنجم درس پردازش گفتار

دکتر ویسی

بهار ۱۴۰۳

فهرست

فهرست شکل‌ها

فهرست جدول‌ها

سوال ۱: پژوهش

۱) فاصله میان توزیع‌ها در یادگیری ماشین

در یادگیری ماشین، اندازه‌گیری فاصله بین دو توزیع به چند دلیل بسیار مهم است، در درجه اول به این دلیل که به درک اینکه دو مجموعه داده چقدر شبیه یا متفاوت هستند کمک می‌کند. این فهم برای کارهایی مانند اعتبارسنجی مدل، تشخیص ناهنجاری، یادگیری انتقالی و موارد دیگر ضروری است.

- یادگیری انتقالی و تطبیق دامنه(Domain Adaptation):

کاربرد: تطبیق مدل ها با دامنه های جدید

در تطبیق دامنه، هدف انطباق مدلی است که در یک دامنه (دامنه منبع) آموزش داده شده است تا به طور موثر در دامنه دیگر (دامنه هدف) کار کند. فاصله بین توزیع دامنه منبع و هدف یک معیار مهم است. اگر توزیع ها بسیار متفاوت باشند، نشان می دهد که مدل ممکن است بدون انطباق، عملکرد خوبی در حوزه هدف نداشته باشد.

مثال:

سناریویی را در نظر بگیرید که در آن یک مدل تجزیه و تحلیل احساسات بر روی نقدهای فیلم (حوزه منبع) آموزش داده شده است، اما باید برای بررسی محصول (دامنه هدف) اعمال شود. توزیع کلمات و احساسات در این دو حوزه ممکن است متفاوت باشد. اندازه گیری فاصله بین این توزیع ها به تعیین میزان سازگاری مورد نیاز کمک می کند.

یک رویکرد رایج، به حداقل رساندن فاصله توزیع با استفاده از تکنیک‌هایی مانند حداکثر اختلاف میانگین (MMD) یا آموزش خصمانه است، که در آن مدل آموزش داده می‌شود تا نه تنها در حوزه منبع خوب عمل کند، بلکه اختلاف توزیع با دامنه هدف را نیز کاهش دهد.

- تشخیص تعمیم‌پذیری مدل

در صورتی که مدل روی داده‌های آموزش و ارزیابی و تست مربوط به یک دیتاست خوب عمل کند، به این معنی نیست که لزوما روی همان وظیفه و روی یک دیتاست یکسان خوب عمل می‌کند، با توجه به عملکرد مدل روی دیتاست‌های مختلف که از توزیع‌های مختلف هستند می‌توان، تعمیم‌پذیری مدل را متوجه شویم.

مثال:

دیتاست‌های تشخیص احساسات صورت معمولا از روی داده‌های آزمایشگاهی (میمیک کردن احساسات به صورت غیر واقعی) ساخته شده‌اند، برای همین روی دیتاست‌هایی که در آن اشخاص احساسات واقعی خود را نشان می‌دهند، لزوما جواب نمی‌دهند.

- تشخیص ناهنجاری

تشخیص ناهنجاری شامل شناسایی الگوهایی در داده هایی است که با رفتار مورد انتظار مطابقت ندارند. مقایسه توزیع داده های جدید با توزیع داده های تاریخی به شناسایی ناهنجاری ها کمک می کند.

مثال:

یک سیستم کشف تقلب مالی را در نظر بگیرید که تراکنش ها را در طول زمان نظارت می کند. سیستم توزیع رفتارهای عادی تراکنش (مقدار، فرکانس، مکان و غیره) را از داده های تاریخی یاد می گیرد.

تشخیص ناهنجاری در زمان واقعی: برای تراکنش های جدید، سیستم فاصله بین توزیع این تراکنش ها و توزیع آموخته شده تراکنش های عادی را اندازه گیری می کند. اگر فاصله زیاد باشد، تراکنش را می توان به عنوان تقلب بالقوه پرچم گذاری کرد.

نظارت بر سیستم: با گذشت زمان، تغییرات در توزیع کلی داده های تراکنش می تواند نشان دهنده تغییر در الگوهای رفتاری باشد که ممکن است نیاز به آموزش مجدد یا به روز رسانی مدل داشته باشد.

چند روش محاسبه فاصله میان دو توزیع:

P و Q دو توزیع با تابع چگالی‌های p و q در نظر می‌گیریم.

۱. **تغییرات کلی :(Total Variation)** فاصله تغییرات کلی بین دو توزیع به صورت زیر تعریف می‌شود:

$$\mathrm{TV}(P,Q) = \sup_A |P(A) - Q(A)| = \sup_A \left| \int \big(p(x) - q(x)\big)\, dx \right|$$

با فرض اینکه A یک زیرمجموعه‌ی قابل اندازه‌گیری از فضای نمونه باشد، فاصله TV می‌سنجد بیشینه‌ی تفاوت بین احتمال رویدادی تحت P نسبت به آن تحت Q است. این فاصله TV معادل فاصله یکی بین چگالی‌ها است، به این معنی که می‌توان نشان داد:

$$\mathrm{TV}(P,Q) = \frac{1}{2} \int |p(x) - q(x)|\ dx$$

انحراف χ^2:

انحراف χ^2 برای توزیع‌های P و Q تعریف می‌شود به گونه‌ای که Q بر P غلبه داشته باشد، به این معنی که اگر Q(A) برای یک مجموعه A برابر با صفر باشد، باید برای P(A) نیز صدق کند که برابر صفر باشد. برای چنین توزیع‌هایی، انحراف χ^2 به صورت زیر تعریف می‌شود:

$$D_{\chi^2}(P|Q) = \int \frac{\big(p(x) - q(x)\big)^2}{q(x)}\, dx$$

در اینجا، p(x) و q(x) به ترتیب نشان‌دهنده‌ی چگالی‌های توزیع‌های P و Q هستند.

انحراف کولبک- لایبلر(Kullback-Leibler Divergence)

دوباره فرض می‌کنیم که Q بر P غلبه دارد. انحراف کولبک-لایبلر بین دو توزیع به صورت زیر تعریف می‌شود :

$$\mathrm{KL}(P,Q) = \int \log\left(\frac{p(x)}{q(x)}\right) p(x)\ dx$$

در اینجا، p(x) و q(x) به ترتیب نشان‌دهنده‌ی چگالی‌های توزیع‌های P و Q هستند و Q باید بر P غلبه داشته باشد.

فاصله هلینگر: فاصله هلینگر بین دو توزیع به صورت زیر تعریف می‌شود:

$$H(P,Q) = \left\{ \int \left(\sqrt{p(x)} - \sqrt{q(x)} \right)^2 dx \right\}^{1/2}$$

به عبارت دیگر، فاصله هلینگر معادل نرم دو بین $\sqrt{q(x)}$ و $\sqrt{p(x)}$ است.

۲) روش نزدیک کردن فاصله دو توزیع

- تبدیل و نرمال‌سازی

یکی یا هر دو توزیع را به منظور بهتر هم‌راستا کردن آنها تغییر دهید. تبدیل‌های رایج شامل:

- **مقیاس‌بندی** :تنظیم مقیاس توزیع‌ها به طوری که دامنه‌های مشابهی داشته باشند.
- **انتقال** :جابجا کردن توزیع‌ها به طوری که میانگین یا میانه آنها هم‌راستا شود.
- **نرمال‌سازی** :تبدیل توزیع‌ها به یک مقیاس مشترک، مانند نمرات استاندارد.(z-scores)

- تکنیک‌های آماری

از تکنیک‌های آماری برای کاهش تفاوت بین توزیع‌ها استفاده کنید:

- **تطابق هیستوگرام** :تنظیم تابع توزیع تجمعی (CDF) یکی از توزیع‌ها به گونه‌ای که با دیگری همخوانی داشته باشد.
- **تبدیل صدکی** :هم‌راستا کردن صدک‌های توزیع‌ها.
- **هموارسازی داده** :اعمال تکنیک‌های هموارسازی برای کاهش نویز و قابل مقایسه‌تر کردن توزیع‌ها.

- روش‌های یادگیری ماشین

از تکنیک‌های یادگیری ماشین برای هم‌راستا کردن توزیع‌ها استفاده کنید:

- **تطابق دامنه** :استفاده از تکنیک‌های تطابق دامنه برای شبیه‌تر کردن داده‌ها از توزیع‌های مختلف.
- **مدل‌های تولیدی** :استفاده از شبکه‌های مولد تخاصمی (GAN) یا کدگشایان متغیر (VAE) برای تولید داده‌ای که با توزیع هدف همخوانی داشته باشد.
- کاهش فاصله آماری

حداقل کردن یک معیار فاصله آماری انتخابی از طریق تکنیک‌های بهینه‌سازی:

- **حداقل کردن واگرایی کولبک-لیبلر (KL)** :بهینه‌سازی پارامترها برای کاهش واگرایی KL بین توزیع‌ها.
- **حداقل کردن فاصله Wasserstein** :استفاده از روش‌های حمل‌ونقل بهینه برای حداقل کردن فاصله.Wasserstein
- **حداقل کردن فاصله جابجایی زمین (EMD)** :هم‌راستا کردن توزیع‌ها با حداقل کردن هزینه تبدیل یک توزیع به دیگری.

- مهندسی ویژگی

تغییر ویژگی‌ها به منظور شبیه‌تر کردن توزیع‌ها:

- **انتخاب ویژگی** :انتخاب ویژگی‌هایی که توزیع‌ها را شبیه‌تر می‌کنند.
- **استخراج ویژگی** :استخراج ویژگی‌های جدید که نمایندگی بهتری از داده‌ها داشته و تفاوت توزیعی را کاهش می‌دهند.

- **تبدیل ویژگی** :اعمال تبدیل‌ها بر روی ویژگی‌های فردی برای هم‌راستا کردن توزیع آنها.

۳) AutoML

یادگیری ماشین (ML) در سال‌های اخیر به موفقیت‌های قابل توجهی دست یافته است و تعداد فزاینده‌ای از رشته‌ها بر آن تکیه می‌کنند. با این حال، این موفقیت به طور اساسی به متخصصان یادگیری ماشین انسانی برای انجام وظایف زیر بستگی دارد:

- پیش‌پردازش و تمیزسازی داده‌ها
- انتخاب ویژگی مناسب
- انتخاب مدل مناسب
- بهینه سازی هایپرپرامترهای مدل
- طراحی توپولوژی شبکه های عصبی (در صورت استفاده از یادگیری عمیق).
- پس‌پردازش مدل‌های یادگیری ماشین
- تجزیه و تحلیل نتایج

از آنجایی که پیچیدگی این وظایف اغلب فراتر از افراد غیرمتخصص در زمینه یادگیری ماشین است، رشد سریع برنامه‌های کاربردی یادگیری ماشین، تقاضایی برای روش های یادگیری ماشینی ایجاد کرده است که افرادی که در این زمینه تخصص ندارند بتوانند از آن استفاده کنند.

Auto MLبه مجموعه‌ای از روش‌ها و ابزارها اطلاق می‌شود که هدف آن‌ها خودکارسازی فرآیندهای مختلف در یادگیری ماشین است. این فرآیندها شامل انتخاب ویژگی‌ها، انتخاب مدل، تنظیم پارامترها، و ارزیابی مدل‌ها می‌شود. Auto ML با هدف کاهش نیاز به تخصص‌های فنی عمیق و افزایش دسترسی به قدرت یادگیری ماشین برای کاربران غیر متخصص توسعه یافته است.

کتابخانه‌های زیادی برای این کار موجود است، تعدادی از آن‌ها به شرح زیر می‌باشند.

- AutoWEKA: یک رویکرد برای انتخاب همزمان الگوریتم یادگیری ماشین و هایپرپارامترهای آن است؛ ترکیب آن با کتابخانه WEKA به طور خودکار مدل‌های خوبی را برای انواع گسترده ای از مجموعه داده ها ارائه می دهد.
- Auto-sklearn: در ادامه‌ی کتابخونه AutoWEKA با استفاده از کتابخانه پایتون scikit-learn است که یک جایگزین قابل استفاده برای طبقه بندی و رگرسیون scikit-learn معمولی است.
- Auto-PyTorch: بر اساس چارچوب یادگیری عمیق PyTorch است و به طور همزمان هایپرپارامترها و معماری عصبی را بهینه می کند.

سایر کتابخانه‌های شناخته شده AutoML عبارتند از:

- AutoGluon: یک رویکرد چندلایه برای ترکیب مدل های مختلف یادگیری ماشین است.
- H2O AutoML: یک پلتفورم برای انتخاب خودکار مدل و ترکیب مدل ها را برای پلتفرم یادگیری ماشین و تحلیل داده ارائه می‌دهد.
- MLBoX: یک کتابخانه AutoML با سه جزء پیش‌پردازش، بهینه‌سازی و پیش‌بینی است.
- TPOT: pipelineهای یادگیری ماشین را با استفاده از ژنتیک بهینه می کند.
- TransmogrifAI: یک کتابخانه AutoML است که بر پایه Spark اجرا می شود.

یکی از روش‌های AutoML در حوزه یادگیری عمیق Neural Architecture Search (NAS) است. NAS به خودکارسازی فرآیند طراحی معماری شبکه‌های عصبی عمیق می‌پردازد .جستجوی معماری عصبی (NAS) یک تکنیک در زمینه یادگیری ماشین خودکار (AutoML) است که هدف آن خودکارسازی طراحی و بهینه سازی معماری های شبکه عصبی برای وظایف خاص است. این روش شامل تعریف یک فضای جستجو از معماری های ممکن و سپس به کارگیری الگوریتم های بهینه سازی برای کاوش و کشف موثرترین معماری در آن فضا است.

نحوه‌ی کارکرد NAS:

ایده اصلی پشت NAS استفاده از الگوریتم های جستجو برای کاوش خودکار یک فضای گسترده از معماری های شبکه عصبی ممکن، ارزیابی عملکرد آنها در یک وظیفه خاص، و شناسایی معماری که بهترین نتایج را به دست می دهد، است. این فرآیند معمولاً شامل مراحل زیر است:

- تعریف فضای جستجو: فضای جستجو دامنه انتخاب‌های معماری را که الگوریتم NAS می تواند کاوش کند، مشخص می‌کند. این شامل انواع لایه‌ها (مانند کانولوشن، پولینگ، متصل کامل)، پیکربندی آن‌ها (مانند اندازه کرنل، گام ها، تابع فعالسازی) و الگوهای اتصال بین لایه‌ها است.
- استراتژی جستجو: استراتژی های جستجوی مختلفی برای پیمایش کارآمد فضای جستجو به کار گرفته می شوند. این موارد شامل یادگیری تقویتی، الگوریتم های تکاملی، بهینه سازی بیزین و روش های مبتنی بر گرادیان است. استراتژی جستجو، کاوش فضای جستجو را هدایت می‌کند، معماری‌های کاندید را تولید می کند و عملکرد آن‌ها را ارزیابی می‌کند.
- تخمین عملکرد: برای ارزیابی عملکرد یک معماری کاندید، باید آن را برای وظیفه هدف آموزش داد و آزمایش کرد. با این حال، آموزش هر معماری از ابتدا می‌تواند از نظر محاسباتی گران باشد. برای رفع این مشکل، الگوریتم‌های NAS اغلب از تکنیک هایی مانند اشتراک وزن، تخمین های کم دقت یا برون یابی منحنی یادگیری برای تقریب عملکرد یک معماری بدون آموزش کامل استفاده می کنند.
- انتخاب معماری: بر اساس تخمین های عملکرد، الگوریتم جستجو به طور تکراری فرآیند جستجو را تعدیل می کند، معماری هایی که عملکرد خوبی دارند را ترجیح می دهد و معماری های با عملکرد ضعیف را کنار می گذارد. هدف این است که به معماری که بهترین عملکرد را برای وظیفه هدف دارد، همگرا شود.

مزایای NAS

- طراحی خودکار معماری NAS : فرآیند زمان بر و دشوار طراحی معماری های شبکه عصبی را که در گذشته به تخصص انسانی و آزمون و خطا متکی است، خودکار می‌کند.
- بهبود عملکرد NAS : پتانسیل کشف معماری هایی را دارد که عملکرد بهتری نسبت به معماری های طراحی شده به صورت دستی برای وظایف خاص داشته باشند، از طریق کاوش یک دامنه گسترده تر از انتخاب های معماری.
- بهینه سازی اختصاصی برای وظیفه NAS : می‌تواند معماری ها را برای وظایف، داده ها و محدودیت های سخت افزاری خاص سفارشی کند، که می تواند منجر به مدل های کارآمدتر و موثرتر شود.
- مقیاس پذیری: با افزایش پیچیدگی وظایف یادگیری عمیق، NAS می‌تواند به مقیاس پذیری فرآیند طراحی کمک کند و با تقاضای معماری‌های پیچیده‌تر گام بردارد.

۴) Kolmogorov–Arnold Networks

شبکه های کلموگروف-آرنولد (KAN) یک معماری جدید از شبکه های عصبی هستند که از قضیه تعبیه کلموگروف-آرنولد الهام گرفته شده اند. این شبکه به عنوان یک جایگزین برای پرسپترون های چند لایه (MLP) سنتی با مزایای بالقوه در دقت، قابلیت تفسیرپذیری و تعداد کمتر پارامتر ارائه می دهند.

قضیه تعبیه کلموگروف-آرنولد بیان می کند که هر تابع پیوسته چندمتغیره می تواند به صورت ترکیب محدودی از توابع پیوسته تک متغیره و عملگر جمع نمایش داده شود. این بینش پیشنهاد می کند که توابع پیچیده چندبعدی می توانند به توابع ساده تر تک متغیره تجزیه شوند.

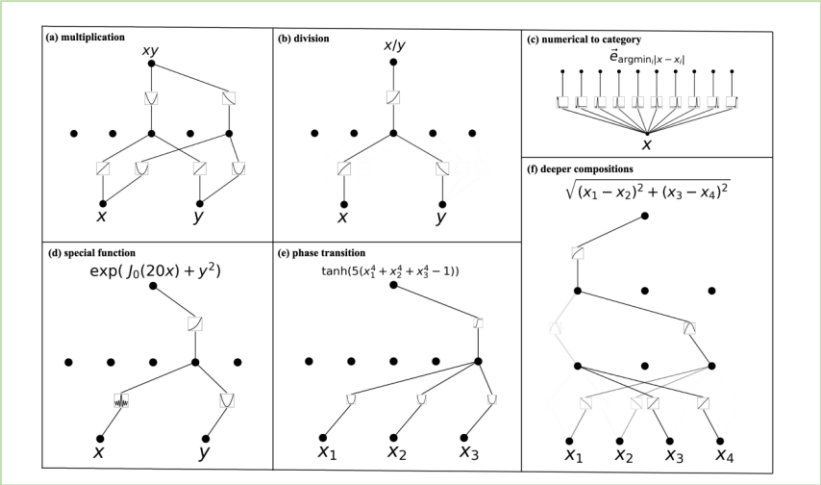
$$[f(x_1, x_2, \dots, x_n) = \sum_{q=1}^{m} \phi_q \left(\sum_{j=1}^{n} \psi_{q,j}(x_j) \right)]$$

KANها از این قضیه با تغییر بنیادی ساختار شبکه های عصبی بهره می برند:

- توابع فعال سازی قابل یادگیری: به جای توابع فعال سازی ثابت که در هر گره اعمال می‌شوند (مانند MLP) KANها توابع فعال سازی قابل یادگیری استفاده می‌کنند. این توابع فعال سازی به عنوان تابع های spline پارامترسازی می شوند، که امکان تنظیم پویا و دقیق را در طول آموزش فراهم می کنند.
- توابع تک متغیره: هر یال در یک KAN یک تابع تک متغیره را نمایش می دهد که روی یک ویژگی ورودی عمل می‌کند. این امر به شبکه اجازه می‌دهد تا مشارکت جداگانه هر ویژگی را به طور جداگانه یاد بگیرد.
- ترکیب و جمع: خروجی این توابع تک متغیره جمع شده و از طریق یک تابع فعال سازی قابل یادگیری دیگر عبور می‌کنند، که در واقع توابع جداگانه را به یک نمایش چندمتغیره ترکیب می کند.

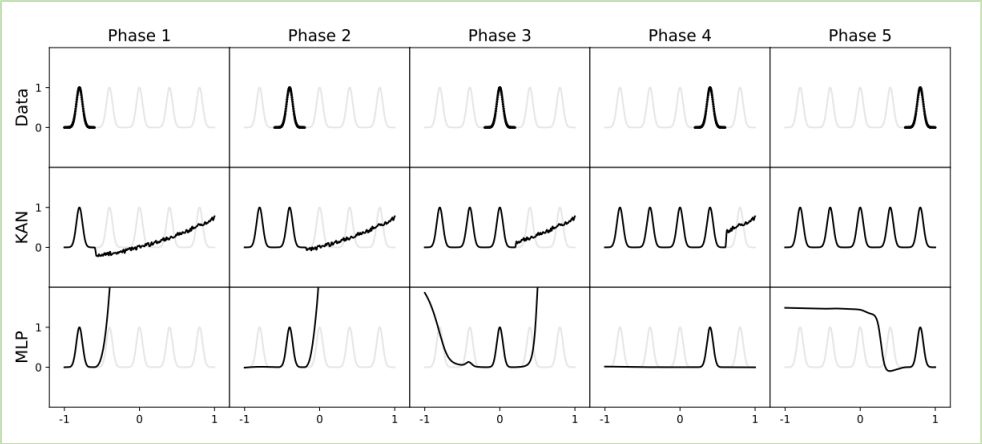
مزایای بالقوه:

- بهبود دقت: با یادگیری پویای توابع فعال سازی، KANها می توانند الگوهای پیچیده در داده‌ها را موثرتر از MLP ها با فعال سازی های ثابت، لحاظ کنند.
- قابلیت تفسیر: امکان مشاهده و درک توابع تک متغیره آموخته شده روی ویژگی های جداگانه می تواند در تفسیر فرآیند تصمیم گیری KAN ها کمک کند، که برای کاربردهایی مانند مراقبت های بهداشتی و امور مالی ضروری است.



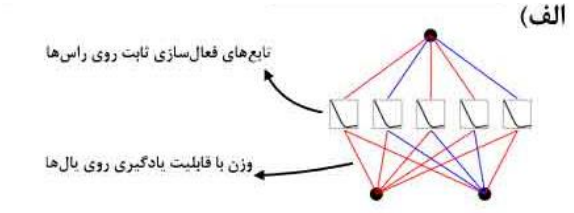
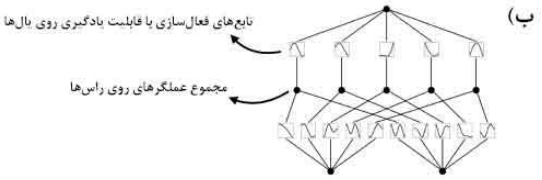
شکل ۱ تفسیرپذیری KAN

- کارایی پارامتر KAN : می‌توانند با تعداد پارامترهای کمتری نسبت به MLP ها، دقت بالاتری را به دست آورند، که آنها را برای انواع مختلف وظایف کارآمدتر و موثرتر می کند.
- جلوگیری از فراموشی: KANها بر خلاف مدل‌های MLP بعد finetune کردن روی وظیفه جدید دچار فراموشی (catastrophic forgetting) نمی‌شوند. کمتر دچار فراموشی ناگهانی می شوند زیرا spline ها کنترل محلی دارند



شکل ۲ مشکل فراموشی در KAN و MLP

جدول مقایسه این دو مدل به شکل زیر می‌باشد:

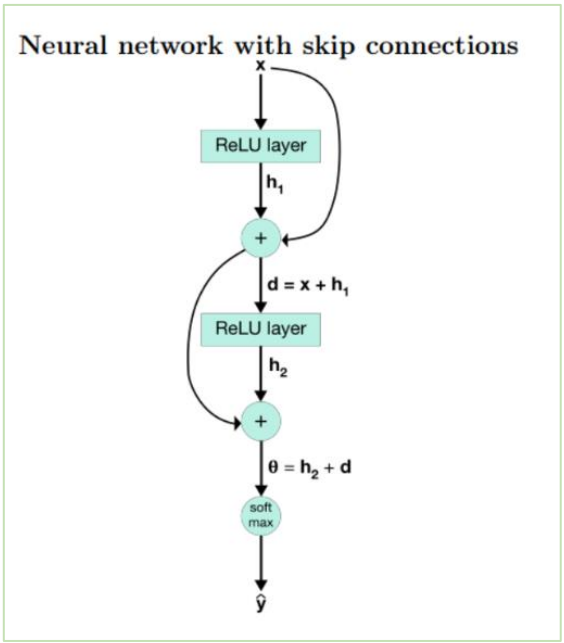
مدل	شبکه عصبی چند لایه (MLP)	شبکه های کلموگروف-آرنولد (KAN)
تئوری مدل	قضیه تقریب جهانی	تئوری تعبیه کلموگروف-آرنولد
فرمول	$[f(x) \approx \sum_{i=1}^{N(e)} a_i \sigma(w_i \cdot x + b_i)]$	$[f(x) = \sum_{q=1}^{2n+1} \Phi_q \left(\sum_{p=1}^{n} \phi_{q,p}(x_p) \right)]$
معماری مدل		
فرمول دقیقتر مدل	$[MLP(x) = (W_3 \circ \sigma_2 \circ W_2 \circ \sigma_1 \circ W_1)(x)]$	$[KAN(x) = (\Phi_3 \circ \Phi_2 \circ \Phi_1)(x)]$

معماری دقیقتر مدل	(پ) MLP(x) W_3 σ_2 W_2 σ_1 W_1 x — غیر خطی ثابت — خطی قابل یادگیری	KAN(x) Φ_3 Φ_2 Φ_1 x — غیر خطی قابل یادگیری

جدول1 مقایسه MLP و KAN

سوال ۲: محو نشویم

معادلات به صورت زیر می‌باشد:



شکل ۳ شبکه عصبی با skip connections

$$z_1 = W_1 x + b_1$$
$$h_1 = \mathrm{ReLU}(z_1)$$
$$d = h_1 + x$$
$$z_2 = W_2 d + b_2$$
$$h_2 = \mathrm{ReLU}(z_2)$$
$$\theta = h_2 + d$$
$$\hat{y} = \mathrm{softmax}(\theta)$$
$$J = \mathrm{CE}(y, \hat{y})$$

الف)

تابع cross-entropy به شکل زیر می‌باشد:

$$L = -\sum_i y_i \log(\widehat{y_i})$$

حال برای محاسبه‌ی گرادیان داریم:

$$\frac{\partial L}{\partial \theta} = \frac{\partial - \sum_i y_i \log(\widehat{y_i})}{\partial \theta} = -\sum_i \log(\widehat{y_i}) \frac{\partial y_i}{\partial \theta} - \sum_i y_i \frac{\partial \log(\widehat{y_i})}{\partial \theta} = -\sum_i y_i \frac{\partial \log(\widehat{y_i})}{\partial \theta}$$
$$= -\sum_i y_i \frac{\partial \log(\widehat{y_i})}{\partial \widehat{y_i}} \cdot \frac{\partial \widehat{y_i}}{\partial \theta} = -\sum_i \frac{y_i}{\widehat{y_i}} \cdot \frac{\partial \widehat{y_i}}{\partial \theta}$$

حال با توجه به مشتق تابع SoftMax داریم:

بنابراین داریم:

$$\frac{\partial L}{\partial \theta} = -\sum_i \frac{y_i}{\widehat{y_i}} \cdot \frac{\partial \widehat{y_i}}{\partial \theta} = -\sum_i \frac{y_i}{\widehat{y_i}} \cdot [\underbrace{\widehat{y_n}(1 - \widehat{y_n})}_{i=n} - \underbrace{\widehat{y_i}\widehat{y_n}}_{i \neq n}] = -\sum_{i=n} \frac{y_i}{\widehat{y_i}} \cdot \widehat{y_n}(1 - \widehat{y_n}) + \sum_{i \neq n} \frac{y_i}{\widehat{y_i}} \cdot \widehat{y_i}\widehat{y_n}$$
$$= -y_n(1 - \widehat{y_n}) + \sum_{i \neq n} y_i \widehat{y_n} = -y_n + \sum_i y_i \widehat{y_n} \xrightarrow{\sum_i y_i = 1} \frac{\partial L}{\partial \theta} = -y_n + \widehat{y_n} = \widehat{y_n} - y_n$$

پس در نهایت داریم:

$$\delta_1 = \frac{\partial J}{\partial \theta} = \frac{\partial J}{\partial \hat{y}} \frac{\partial \hat{y}}{\partial \theta} = \hat{y} - y$$

ب)

$$\delta_2 = \frac{\partial J}{\partial z_2} = \frac{\partial J}{\partial \theta} \frac{\partial \theta}{\partial h_2} \frac{\partial h_2}{\partial z_2} = (\hat{y} - y) \cdot 1 \cdot \underbrace{\frac{\partial \mathrm{ReLU}(z_2)}{\partial z_2}}_{\circ 1\{z_2 > 0\}} = \delta_1 \circ 1\{z_2 > 0\}$$

ج)

$$\delta_3 = \frac{\partial J}{\partial d} = \underbrace{\frac{\partial J}{\partial \theta} \frac{\partial \theta}{\partial h_2} \frac{\partial h_2}{\partial z_2}}_{\delta_2} \underbrace{\frac{\partial z_2}{\partial d}}_{W_2^T} + \underbrace{\frac{\partial J}{\partial \theta}}_{\delta_1} \underbrace{\frac{\partial \theta}{\partial d}}_{1} = W_2^T \delta_2 + \delta_1$$

د)

$$\frac{\partial J}{\partial x} = \frac{\partial J}{\partial d} \frac{\partial d}{\partial h_1} \frac{\partial h_1}{\partial x} + \underbrace{\frac{\partial J}{\partial d}}_{\delta_3} \underbrace{\frac{\partial d}{\partial x}}_{1} = W_1^T(\delta_3 \circ 1\{z_1 > 0\}) + \delta_3$$

سوال ۳: نجوا کرم یا نجوا گرم ؟

الف) تبدیل صوت فارسی به متن

جمله‌ی "سلام، من دیروز در کلاس پردازش گفتار دیجتال شرکت کردم" را به مدل می‌دهیم، مدل Whisper base پیش‌بینی زیر را برای این صوت انجام می‌دهد:

```
Detected language: fa
Transcription:  سلام، من تیروز، در کلاس، پر دازش قفتار دیشیطال شرکت کردام
```

که تقریبا تبدیل به متن درستی انجام داده است اما در واجهای نزدیک بهم اشتباهاتی وجود دارد.

ب) ترجمه صوت چینی به انگلیسی

جمله‌ی "? hello, how're you doing" به زبان چینی یعنی "？你好，近来怎么样" را به مدل می‌دهیم، مدل Whisper base پیش‌بینی زیر را برای این صوت انجام می‌دهد:

```
Hello, how are you doing?
```

که ترجمه‌ی درستی می‌باشد.

پ) یادگیری انتقالی

یادگیری انتقالی (Transfer Learning) یک روش پرکاربرد در علم یادگیری ماشین است که از دانش یک مدل آموزش دیده بر روی یک مسئله، برای بهبود عملکرد یک مسئله مشابه دیگر استفاده می‌کند. عموماً در یادگیری انتقالی، مدلی که بر روی یک مجموعه داده‌های منبع مانند مجموعه داده‌های ImageNet آموزش دیده است، به عنوان مدل پایه استفاده می‌شود و برای مسئله مقصد، که ممکن است دارای مجموعه داده‌های کمتری باشد، به‌روزرسانی می‌شود. به عبارت دیگر در یادگیری انتقال ما مدلی که روی یک مجموعه داده‌ی بزرگ آموزش دیده است و می‌تواند با دقت خوبی پاسخ بدهد، (برای مثال در شبکه‌های عصبی کانولوشنالی، به این معنی است که ویژگی‌های مناسبی در لایه‌های اول استخراج می‌شود) روی مجموعه داده‌ای که برای وظیفه‌ی مشابه هست آموزش می‌دهیم و به جای اینکه از صفر مدل شروع به آموزش کند مقداری در انجام این وظیفه قبلا آموزش دیده است.

مزیت‌های یادگیری انتقالی:

۱. **صرفه‌جویی در زمان و منابع محاسباتی:**
   - از آنجا که مدل‌های پیش‌آموزش دیده شده از قبل وجود دارند، نیاز به آموزش مدل از ابتدا کاهش می‌یابد. این باعث صرفه‌جویی در زمان و منابع محاسباتی می‌شود.
۲. **بهبود عملکرد با داده‌های کم:**

- یادگیری انتقالی به مدل‌ها اجازه می‌دهد که با داده‌های کم‌تر نیز عملکرد خوبی داشته باشند، چرا که مدل از اطلاعاتی که در وظایف قبلی آموخته استفاده می‌کند.

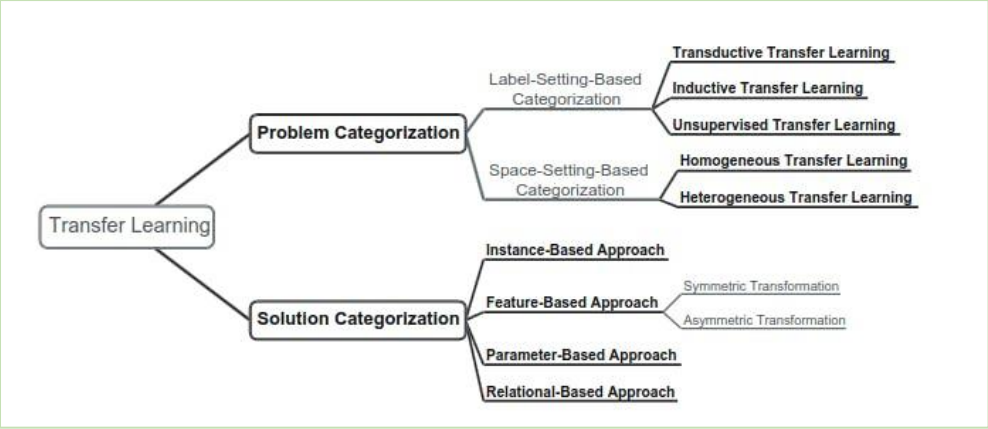
۳. **تسریع در فرایند توسعه مدل:**

- استفاده از مدل‌های پیش‌آموزش دیده شده به توسعه‌دهندگان امکان می‌دهد که سریع‌تر به نتایج مطلوب برسند و فرایند آزمایش و خطا کاهش یابد.

٤. **کاهش نیاز به داده‌های برچسب‌دار:**

- با استفاده از یادگیری انتقالی، نیاز به برچسب‌گذاری داده‌ها به مقدار زیادی کاهش می‌یابد زیرا مدل‌های پیش‌آموزش دیده شده از قبل دانش عمومی مناسبی دارند.

دسته‌بندی‌های یادگیری انتقالی



شکل ۴ دسته‌بندی‌های یادگیری انتقالی

- یادگیری انتقالی استقرایی (Inductive Transfer Learning)

یادگیری انتقالی استقرایی تکنیکی است که زمانی استفاده می‌شود که داده‌های برچسب‌گذاری‌شده بین حوزه‌های منبع و هدف سازگار باشند، اما وظایفی که توسط مدل‌ها انجام می‌شود متفاوت است. این تکنیک شامل انتقال دانش بین وظایف یا حوزه‌ها است. هنگامی که انتقال بین وظایف صورت می‌گیرد، درک مدل از یک وظیفه به حل وظیفه‌ای متفاوت اما مرتبط کمک می‌کند. به عنوان مثال، استفاده از مدلی که بر روی طبقه‌بندی تصاویر آموزش دیده است، عملکرد تشخیص اشیاء را بهبود می‌بخشد. انتقال بین حوزه‌ها این مفهوم را به مجموعه داده‌های مختلف گسترش می‌دهد. به عنوان مثال، مدلی که ابتدا بر روی عکس‌های حیوانات آموزش دیده است، می‌تواند برای تحلیل تصاویر پزشکی بهینه شود.

- یادگیری انتقالی تراگرد (Transductive Transfer Learning)

در یادگیری تراگرد، مدل از قبل با داده‌های آموزش و آزمون مواجه شده است. با یادگیری از مجموعه داده‌های آموزش آشنا، یادگیری تراگرد پیش‌بینی‌هایی را بر روی مجموعه داده‌های آزمون انجام می‌دهد. در حالی که برچسب‌های مجموعه داده‌های آزمون ممکن است ناشناخته باشند، مدل از الگوهای آموخته‌شده خود برای پیشبرد فرآیند پیش‌بینی استفاده می‌کند.

یادگیری انتقالی تراگرد در سناریوهایی اعمال می‌شود که حوزه‌های وظایف منبع و هدف شباهت زیادی دارند اما دقیقا یکسان نیستند. به عنوان مثال، مدلی که برای طبقه‌بندی انواع مختلف گل‌ها از تصاویر برچسب‌گذاری‌شده آموزش دیده است (حوزه منبع). وظیفه هدف، شناسایی گل‌ها در نقاشی‌های هنری بدون برچسب است (حوزه هدف). در اینجا، مدل از توانایی‌های آموخته‌شده خود در شناسایی گل‌ها از تصاویر برچسب‌گذاری‌شده برای پیش‌بینی انواع گل‌های موجود در نقاشی‌ها استفاده می‌کند.

حال اگر منظور از سوال استراتژی‌های یادگیری انتقالی بود، آن‌ها به شرح زیر می‌باشند:

استراتژی انتقال یادگیری انتقالی نیازمند یافتن مسیر مناسب برای تطبیق دانش مدل است. در اینجا سه استراتژی متمایز را برای در نظر گرفتن، با توجه به سناریوهای مختلف و دسترسی به داده‌ها، معرفی می‌کنیم.

- تنظیم کامل: این رویکرد از داده‌های هدف برای تنظیم کامل مدل استفاده می‌کند. این روش زمانی مؤثر است که مقدار قابل توجهی داده‌های آموزشی برچسب‌گذاری‌شده برای وظیفه هدف موجود باشد.
- تنظیم لایه‌به‌لایه: این استراتژی شامل تنظیم لایه‌های خاصی است تا تخصص مدل پیش‌آموزش‌دیده را تطبیق دهد. این رویکرد زمانی مناسب است که داده‌های هدف محدود باشد.
- استخراج ویژگی: در این روش، لایه‌های پیش‌آموزش‌دیده ثابت نگه داشته می‌شوند و ویژگی‌های آموخته‌شده از آن‌ها استخراج می‌گردد. مدل جدید بر اساس ویژگی‌های آموخته‌شده برای وظیفه پایین‌دستی آموزش داده می‌شود. این روش زمانی خوب عمل می‌کند که مجموعه داده هدف کوچک باشد. مدل جدید از دانش عمومی لایه‌های پیش‌آموزش‌دیده بهره‌مند می‌شود.

ج) تنظیم دقیق مدل **Whisper** برای مجموعه‌داده فارسی

ابتدا داده‌ها را طبق خواسته‌ی سوال، داده‌ها را بارگیری می‌کنیم، همچنین رونوشت شده‌ی صوت‌ها و متن واقعی آن‌ها را نیز در یک جدول ذخیره می‌کنیم.

	audio	transcription	real transcription
0	myaudio/450001.wav	... بدام خودام در سیلو که در خدمت شما خواهیم بود	...یه نام خدا درسی رو که در خدمت شما خواهیم بود آ
1	myaudio/450002.wav	...دوعو کار کردن با لنک و ایجاد مستندات و مرجعها	... در واقع کار کردن با لنک و ایجاد مستندات و متن
2	myaudio/450003.wav	اینگلیسی و کلان مستندات اینلمی هدافی نزک تجایی که	...انگلیسی و کلا مستندات علمی هدف این است که تا ج
3	myaudio/450004.wav	...ممکنه مطالبی روگی بیشتر کرای داره برا تون بدا	...که ممکنه مطالبی رو که بیشتر کارایی داره براتون
4	myaudio/450005.wav	...آموزش داده بشه و مجموه خوبی در اختیار شما عرا	...آموزش داده بشه و یه مجموعه ی خوبی در اختیار شم

شکل ۵ رونوشت مجموعه‌داده

حال داده‌ها نرخ خطای داده‌ها را بدست می‌آوریم که به شرح زیر می‌باشد:

```
Initial WER for the train data: 70.49%
Initial WER for val_data: 72.64%
```

```
Initial WER for test_data: 70.44%
Initial WER for all the data: 70.83%
```

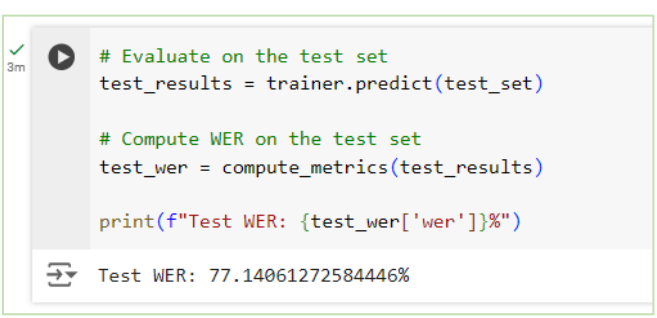
البته، خطاهای بالا حاصل از دو بخش تشخیص زبان و تشخیص گفتار است(به عبارت دیگر اگر در تشخیص زبان مدل اشتباه کند به صورت آبشاری این اشتباه به تشخیص گفتار نیز انتقال می‌یابد).

حال بعد از آماده کردن داده‌ها و تبدیل آن‌ها به dataloader مورد نظر، مدل Whisper را با کمک api هاگینگ فیس برای ۵ epoch و با نرخ آموزش 1e-5 برای شروع آموزش می‌دهیم، همچنین برای هر ۵۰ گام مدل را روی داده‌های ارزیابی بررسی می‌کنیم.

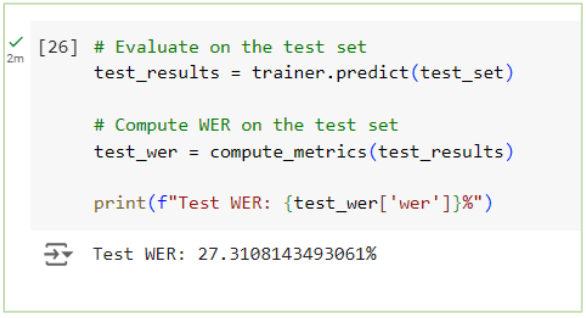
Step	Training Loss	Validation Loss	Wer
50	0.526500	0.440869	37.927726
100	0.214700	0.332150	29.357211
150	0.095000	0.328409	29.804925
200	0.051300	0.326172	28.813559

جدول ۲ عملکرد Whisper حین آموزش

پس از پایان آموزش، wer مدل روی داده‌های تست قبل و بعد آموزش به صورت زیر است:



شکل 7 WER قبل از آموزش

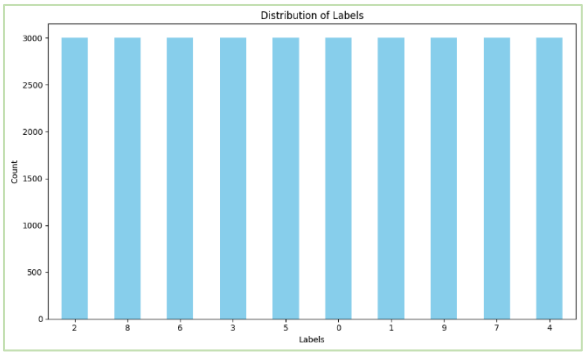


شکل ۶ WER بعد از آموزش

مشخصا مدل به شدت بعد از ۵ دوره آموزش به شدت بهتر شده است، همچنین اگر تعداد دوره‌ها زیاد کنیم مدل همچنان بهتر خواهد شد، اما به واسطه ضعیف بودن سیستم و طول کشید از این کار صرف نظر شده است.

سوال ۴: پیاده‌سازی شبکه LSTM

ابتدا مجموعه داده را از سایت کگل دانلود می‌کنیم و به بررسی‌های اولیه می‌پردازیم. توزیع داده‌ها در همه‌ی کلاس‌ها یکسان است، بنابراین نیاز به پیش‌پردازش اولیه‌ای در این بخش نداریم.



شکل ۸ توزیع داده‌های مجموعه‌داده‌ی AudioMNIST

گام ۱: فراخوانی داده‌ها و بخش‌بندی

داده‌ها را به نسبت ۸۰ و ۲۰ به دو دسته‌ی آموزش و آزمون تقسیم می‌کنیم. برای اینکار از کد زیر استفاده می‌کنیم:

```
X_train, X_test, Y_train, Y_test = train_test_split(features, Y, test_size=0.2, random_state=42)
```

البته بدون استفاده از کتابخانه هم این کار ممکن است که تابع آن نیز در کد تعریف شده است.

گام ۲: پیش‌پردازش‌های لازم روی داده‌ها

در این بخش دو روش برای استخراج ویژگی استفاده کرده‌ایم:

- تابع extract_mfcc با در نظر گرفتن طول فریم ۲۵ میلی‌ثانیه و ۱۳ ویژگی MFCC، ویژگی‌های MFCC را استخراج می‌کند. و تمام ویژگی‌ها را connate می‌کنیم. نتایج ذکر شده از در بخش‌های آتی، حاصل از این نوع استخراج ویژگی می‌باشد.
- همچنین، علاوه بر روش ذکر شده، حالتی که از تمام mfccها میانگین بگیریم و فقط ۱۳ ویژگی داشته باشیم نیز به عنوان روش استخراج ویژگی استفاده شده است که در کد نتایج آن موجود است و در این گزارش ذکر نشده است.

گام ۳: ساخت و آموزش مدل

۱. **ورودی ویژگی‌ها:**
   - ویژگی‌های دادگان به عنوان ورودی به مدل داده می‌شوند.
   - این ویژگی‌ها به یک لایه LSTM با ۶۴ واحد مخفی (hidden state) ارسال می‌شوند.
۲. **لایه LSTM :**
   - خروجی لایه اول LSTM به یک لایه دیگر LSTM با ۶۴ واحد مخفی داده می‌شود.
۳. **لایه‌های Dense :**
   - خروجی لایه دوم LSTM به یک لایه Dense با ۶۴ نورون و تابع فعال‌ساز ReLU داده می‌شود.
   - پس از لایه Dense، یک لایه Dropout با نرخ ۰.۳ قرار داده می‌شود.
   - در نهایت، یک لایه Dense با ۱۰ نورون خروجی و تابع فعال‌ساز Softmax برای طبقه‌بندی داده‌ها به ۱۰ کلاس مختلف قرار داده می‌شود.
۴. **بهینه‌سازی و تنظیمات مدل:**
   - از بهینه‌ساز Adam با نرخ یادگیری (learning rate) برابر با ۰.۰۰۰۱ استفاده شده است.

- تابع خسارت (loss function) مدل، sparse_categorical_crossentropy در نظر گرفته شده است.

آموزش مدل

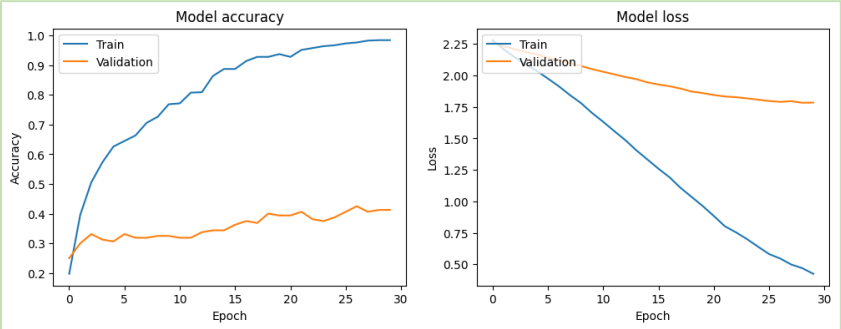
مدل با تنظیمات زیر آموزش داده شد:

- تعداد epoch: ۳۰
- اندازه batch: ۳۲

در هر epoch ، دقت (accuracy) مدل برای داده‌های آموزشی و ارزیابی گزارش شده است.

نتایج به شرح زیر می‌باشد:

نمودار خطا و دقت در طول آموزش به شکل زیر می‌باشد:

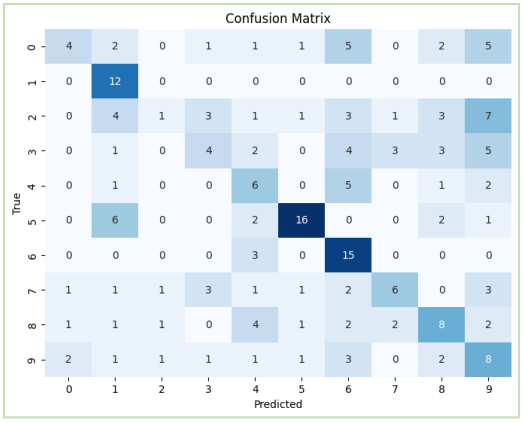


شکل ۹ نمودار دقت و خطا در طول آموزش برای LSTM با دو لایه ۶۴ نورونی

ماتریس درهمریختگی و نتایج مدل روی داده‌های تست به شرح زیر می‌باشد:

Classification Report:

	precision	recall	f1-score	support
0	0.50	0.19	0.28	21
1	0.41	1.00	0.59	12
2	0.25	0.04	0.07	24
3	0.33	0.18	0.24	22
4	0.29	0.40	0.33	15
5	0.76	0.59	0.67	27
6	0.38	0.83	0.53	18
7	0.50	0.32	0.39	19
8	0.38	0.36	0.37	22
9	0.24	0.40	0.30	20
accuracy			0.40	200
macro avg	0.41	0.43	0.38	200
weighted avg	0.42	0.40	0.37	200



شکل 10 عملکرد مدل LSM با دو لایه ۶۴ نورونی روی داده‌های Test

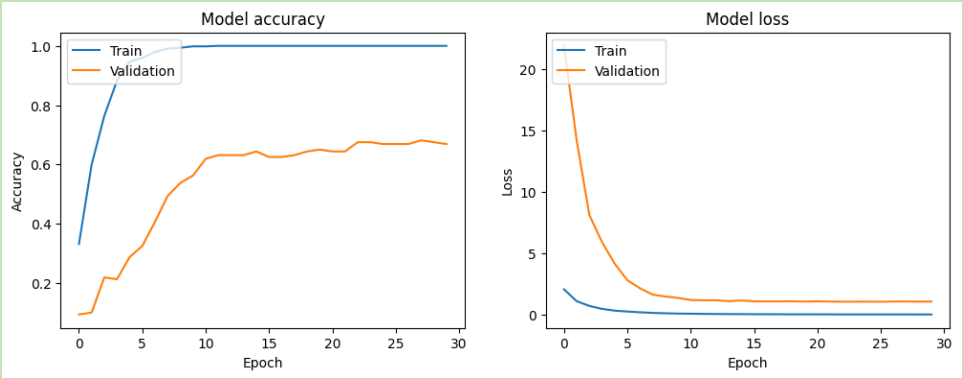
دقت و خطای مدل برای داده‌های تست به شرح زیر می‌باشد:

```
7/7 [==============================] - 0s 10ms/step - loss: 1.8318 - accuracy: 0.4000
```

این مدل دقت ۴۰ روی داده‌های تست می‌دهد که از حالت انتخاب تصادفی به شدت بهتر است اما باز می‌تواند بهتر شود. دلایل کم بودن دقت این مدل می‌تواند به دلایل زیر باشد:

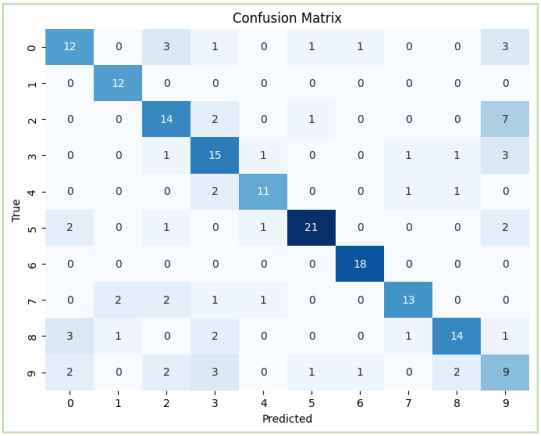
- ضعیف بودن مدل
  - مشخصا با اضافه کردن لایه‌های بیشتر یا hidden state های بیشتر مدل بهتر عمل خواهد کرد، همانطور که در بخش بعدی همین موضوع را خواهیم دید.
  - **وابستگی‌های زمانی کوتاه‌مدت** LSTM: برای مدل‌سازی وابستگی‌های زمانی طولانی‌مدت مناسب هستند. در حالی که در پردازش ویژگی‌های MFCC که در بازه‌های زمانی کوتاه استخراج می‌شوند، ممکن است وابستگی‌های زمانی طولانی‌مدت چندان مهم نباشند.
- ویژگی‌های بیشتر و داده‌های بیشتر
  - با افزایش داده‌ها مدل بهتر عمل خواهد کرد و این داده‌ها کم می‌باشند.
  - ویژگی‌های mfcc
    - تنها ۱۳ ویژگی برای هر فریم زمانی دارند، که ممکن است اطلاعات کافی برای تشخیص ژانرهای موسیقی را فراهم نکند.
    - فشرده‌سازی اطلاعات: MFCCها به نوعی فشرده‌سازی اطلاعات صوتی محسوب می‌شوند و ممکن است برخی اطلاعات مهم فرکانسی را از دست بدهند.
    - حساسیت به نویز: ویژگی‌های MFCC ممکن است به نویز و تغییرات جزئی در سیگنال حساس باشند.
- Overfit شدن: مشخصا مدل به داده‌های آموزش overfit شده است، برای حل این مشکل می‌توانیم از تکنیک‌های جلوگیری از overfitting مانند، drop out این مشکل را حل کنیم.

همچنین نتایج مدل CNN به شرح زیر می‌باشد:



Classification Report:

	precision	recall	f1-score	support
0	0.63	0.57	0.60	21
1	0.80	1.00	0.89	12
2	0.61	0.58	0.60	24
3	0.58	0.68	0.62	22
4	0.79	0.73	0.76	15
5	0.88	0.78	0.82	27
6	0.90	1.00	0.95	18
7	0.81	0.68	0.74	19
8	0.78	0.64	0.70	22
9	0.36	0.45	0.40	20
accuracy			0.69	200
macro avg	0.71	0.71	0.71	200
weighted avg	0.71	0.69	0.70	200



شکل ۱۱ نتایج مدل CNN

مشخصا مدل CNN نسبت به این مدل بهتر عمل می‌کند، این موضوع چند دلیل می‌تواند داشته باشد.

- MFCC ها نمایش فشرده تری از اطلاعات طیفی هستند که با گرفتن تبدیل کسینوس گسسته (DCT) طیف log mel به دست می آیند. در حالی که این ویژگی‌ها را به هم مرتبط می‌کند و آنها را برای مدل‌سازی آماری مناسب‌تر می‌کند، برخی از اطلاعات فضایی موجود در طیف‌نگار mel را نیز دور می‌اندازد. بنابراین mel ویژگی‌های بهتری را دارا است، علاوه بر این مدلی که بتواند mfccها را به خوبی و بدون overfit شدن یاد بگیرد نیاز به پیچیدگی بیشتری دارد ولی اینجا مدل ما ساده می‌باشد.
- تشخیص الگوهای مکانی: CNNها به خوبی قادر به تشخیص الگوهای مکانی و فرکانسی در تصاویر هستند. Mel-Spectrogram یک نمایش تصویری از سیگنال صوتی است و CNNها می‌توانند الگوهای فرکانسی مختلف را در طول زمان به خوبی تشخیص دهند.
- تعداد پارامترها: CNNها به دلیل استفاده از لایه‌های کانولوشنی تعداد پارامترهای زیادی دارند که باعث می‌شود مدل توانایی یادگیری پیچیدگی‌های بیشتری را داشته باشد.

در نهایت ترکیب LSTM با لایه‌های کانولوشنالی احتمالا که هم الگوهای کوتاه مدت و هم بلند مدت را پیدا می‌کند، بهترین جواب را خواهد داد.

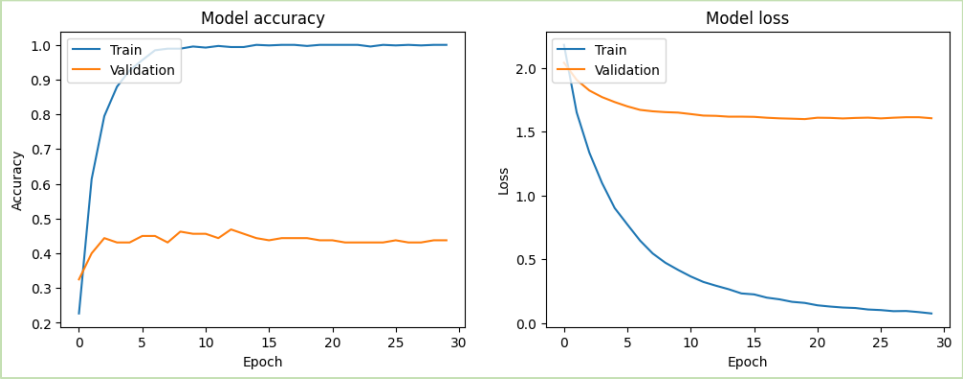
بخش امتیازی

الف)

مدل LSTM با یک لایه با hidden state۱۲۸

برای این بخش، مدل LSTM با یک لایه و ۱۲۸ hidden state طراحی شده و بر روی داده‌های MFCC آموزش داده می‌شود.
در اینجا فرض می‌کنیم که تمام پیش‌فرض‌های قبلی مانند learning rate، batch size یکسان هستند.

نمودار خطا و دقت در طول آموزش به شکل زیر می‌باشد:

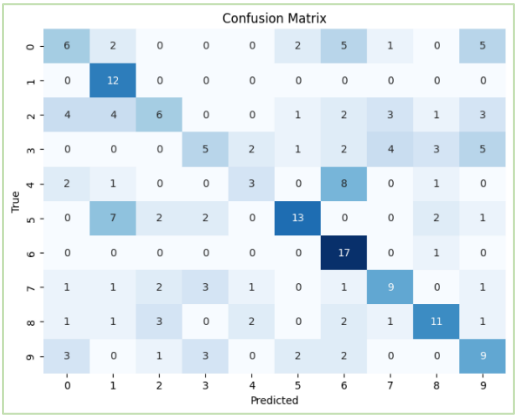


شکل ۱۲ نمودار دقت و خطا در طول آموزش برای LSTM با یک لایه ۱۲۸ نورونی

ماتریس درهمریختگی و نتایج مدل روی داده‌های تست به شرح زیر می‌باشد:

Classification Report:

	precision	recall	f1-score	support
0	0.35	0.29	0.32	21
1	0.43	1.00	0.60	12
2	0.43	0.25	0.32	24
3	0.38	0.23	0.29	22
4	0.38	0.20	0.26	15
5	0.68	0.48	0.57	27
6	0.44	0.94	0.60	18
7	0.50	0.47	0.49	19
8	0.58	0.50	0.54	22
9	0.36	0.45	0.40	20
accuracy			0.46	200
macro avg	0.45	0.48	0.44	200
weighted avg	0.46	0.46	0.43	200



شکل ۱۳عملکرد مدل LSM با یک لایه ۱۲۸ نورونی روی داده‌های Test

دقت و خطای مدل برای داده‌های تست به شرح زیر می‌باشد:

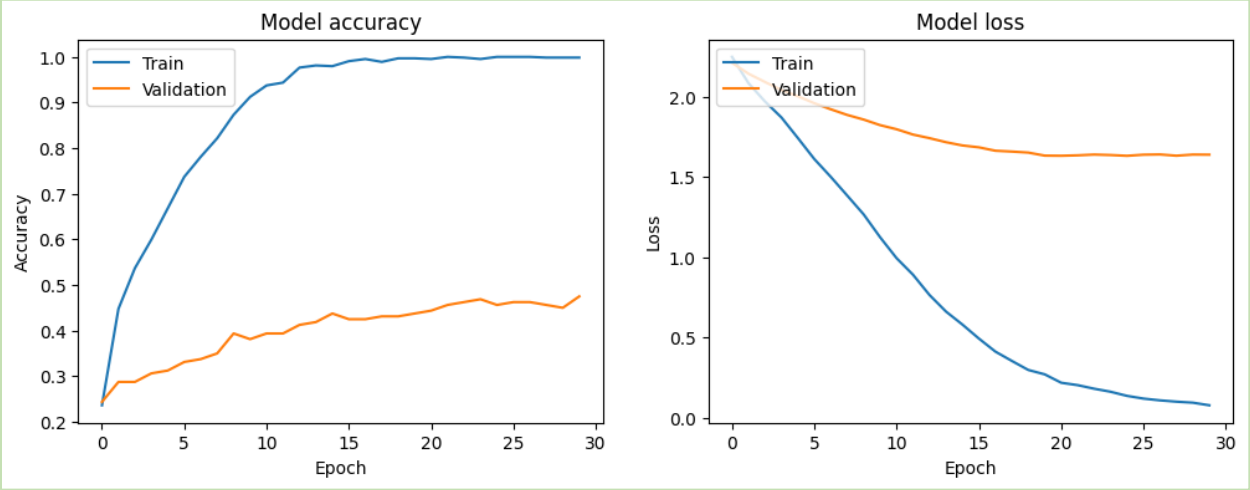
```
7/7 [==============================] - 1s 15ms/step - loss: 1.6996 - accuracy: 0.4550
```

- این مدل عملکرد بهتری دارد، که ممکن است به دلایل زیر باشد:
- پارامترهای کمتر: با استفاده از یک لایهLSTM ، تعداد پارامترها کمتر می‌شود که ممکن است باعث کاهش نیاز به محاسبات و زمان آموزش شود. و از آنجا که مدل قبلی overfit شده بود، مدل با پارامترهای کمتر مشخصا نتیجه بهتری روی داده‌های تست می‌دهد.
- مدل‌سازی وابستگی‌های زمانی کوتاه‌تر: یک لایه LSTM با تعداد واحدهای بیشتر ممکن است به خوبی بتواند وابستگی‌های زمانی کوتاه‌تر بهتری نسبت به دو لایه با تعداد واحدهای کمتر مدل‌سازی کند.

ب)

۱- مدلBiLSTM با دو لایه ۶۴ نورونی

برای این بخش، مدل BiLSTM با دو لایه ۶۴ نورونی و بر روی داده‌های MFCC آموزش داده می‌شود. در اینجا فرض می‌کنیم که تمام پیش‌فرض‌های قبلی مانند learning rate، batch size یکسان هستند. مدل را با توجه به خواسته‌ی سوال می‌سازیم و آموزش می‌دهیم، نتایج به شکل زیر می‌باشد:



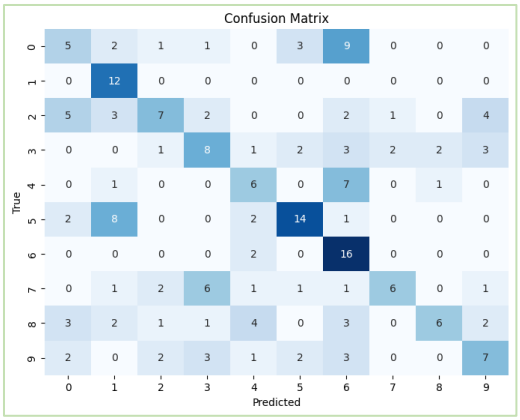
شکل ۱۴ نمودار دقت و خطا در طول آموزش برای BILSTM با دو لایه ۶۴ نورونی

ماتریس درهمریختگی و نتایج مدل روی داده‌های تست به شرح زیر می‌باشد:

```
Classification Report:
              precision    recall  f1-score   support

           0       0.29      0.24      0.26        21
           1       0.41      1.00      0.59        12
           2       0.50      0.29      0.37        24
           3       0.38      0.36      0.37        22
           4       0.35      0.40      0.38        15
           5       0.64      0.52      0.57        27
           6       0.36      0.89      0.51        18
           7       0.67      0.32      0.43        19
           8       0.67      0.27      0.39        22
           9       0.41      0.35      0.38        20

    accuracy                           0.43       200
   macro avg       0.47      0.46      0.42       200
weighted avg       0.48      0.43      0.42       200
```



شکل ۱۵ عملکرد مدل BILSTM با دو لایه ۶۴ نورونی روی داده‌های Test

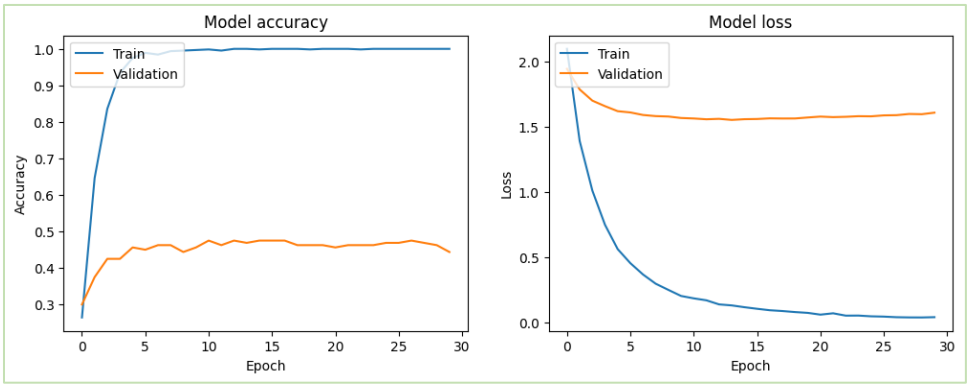
دقت و خطای مدل برای داده‌های تست به شرح زیر می‌باشد:

```
7/7 [==============================] - 2s 19ms/step - loss: 1.8226 - accuracy: 0.4350
```

این مدل نسبت به LSTM دو لایه با ۶۴ نورون بهتر عمل می‌کند. استفاده از Bidirectional LSTM به دلیل استفاده از اطلاعات زمانی هم‌زمان از قبل و بعد از هر نقطه، بهبودی کوچک‌تری در دقت نسبت به Unidirectional LSTM داشته است. این نشان می‌دهد که مدل با توانایی بالاتر در مدل‌سازی وابستگی‌های زمانی می‌تواند دقت بهتری را داشته باشد.

۲- مدل BiLSTM با یک لایه ۱۲۸ نورونی

برای این بخش، مدل BiLSTM با یک لایه ۱۲۸ نورونی و بر روی داده‌های MFCC آموزش داده می‌شود. در اینجا فرض می‌کنیم که تمام پیش‌فرض‌های قبلی مانند learning rate، batch size یکسان هستند. مدل را با توجه به خواسته‌ی سوال می‌سازیم و آموزش می‌دهیم، نتایج به شکل زیر می‌باشد:



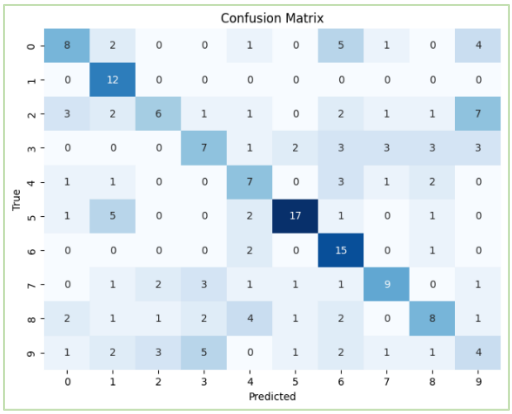
شکل ۱۶ نمودار دقت و خطا در طول آموزش برای BILSTM با یک لایه ۱۲۸ نورونی

ماتریس درهمریختگی و نتایج مدل روی داده‌های تست به شرح زیر می‌باشد:

```
Classification Report:
              precision    recall  f1-score   support

           0       0.50      0.38      0.43        21
           1       0.46      1.00      0.63        12
           2       0.50      0.25      0.33        24
           3       0.39      0.32      0.35        22
           4       0.37      0.47      0.41        15
           5       0.77      0.63      0.69        27
           6       0.44      0.83      0.58        18
           7       0.56      0.47      0.51        19
           8       0.47      0.36      0.41        22
           9       0.20      0.20      0.20        20

    accuracy                           0.47       200
   macro avg       0.47      0.49      0.46       200
weighted avg       0.48      0.47      0.45       200
```



شکل ۱۷ عملکرد مدل BILSTM با یک لایه ۱۲۸ نورونی روی داده‌های Test

دقت و خطای مدل برای داده‌های تست به شرح زیر می‌باشد:

```
7/7 [==============================] - 3s 43ms/step - loss: 1.7337 - accuracy: 0.4650
```

این مدل نسبت به LSTM یک لایه با ۱۲۸ نورون بهتر عمل می‌کند. استفاده از Bidirectional LSTM به دلیل استفاده از اطلاعات زمانی هم‌زمان از قبل و بعد از هر نقطه، بهبودی کوچک‌تری در دقت نسبت به Unidirectional LSTM داشته است. این نشان می‌دهد که مدل با توانایی بالاتر در مدل‌سازی وابستگی‌های زمانی می‌تواند دقت بهتری را داشته باشد.