دانشگاه صنعتی امیرکبیر
(پلی تکنیک تهران)

دانشکده مهندسی برق

کنترل و عیب‌یابی هوشمند

پیاده‌سازی روش‌های هوشمند برای کنترل و عیب‌یابی سیستم وسیله‌ی زیرآبی خودران (AUV)

نگارش

علی نظامی وند چگینی

استاد

دکتر فرزانه عبدالهی

بهار 1404

چکیده

مقدمه

در سال‌های اخیر، وسایل زیرآبی خودران (AUVs) به عنوان یکی از ابزارهای پیشرفته در حوزه‌های دریایی، نظامی، صنعتی و تحقیقاتی، جایگاه ویژه‌ای یافته‌اند. این وسایل بدون نیاز به کنترل مستقیم انسانی، قادرند مأموریت‌های پیچیده‌ای همچون نقشه‌برداری از بستر دریا، پایش زیست‌محیطی، بازرسی زیرساخت‌های دریایی و عملیات جست‌وجو و نجات را با دقت و بهره‌وری بالا انجام دهند. با گسترش کاربردهای AUV و افزایش پیچیدگی مأموریت‌ها، نیاز به کنترل دقیق، پایدار و مقاوم در برابر اغتشاشات و خطاهای عملکردی این سیستم‌ها بیش از پیش احساس می‌شود.

از سوی دیگر، به دلیل عملکرد مستقل و حضور طولانی‌مدت در محیط‌های ناشناخته و متغیر زیرآبی، سیستم‌های AUV در معرض انواع خطاهای سنسوری، عملگری و ساختاری قرار دارند. شناسایی سریع و دقیق این خطاها، و اعمال اقدامات کنترلی مناسب به منظور حفظ پایداری و ایمنی سیستم، از جمله چالش‌های اساسی در توسعهAUV های پیشرفته به شمار می‌رود.

در این راستا، بهره‌گیری از روش‌های **کنترل و عیب‌یابی هوشمند** مبتنی بر یادگیری ماشین، شبکه‌های عصبی، منطق فازی و الگوریتم‌های تطبیقی، راهکاری مؤثر جهت افزایش قابلیت اطمینان، انعطاف‌پذیری و عملکرد این سیستم‌ها فراهم می‌سازد. هدف این مقاله، بررسی و توسعه چارچوبی هوشمند برای کنترل و شناسایی عیب درAUV ها با تکیه بر مدل‌سازی دقیق دینامیکی، طراحی کنترلرهای مقاوم و بهره‌گیری از ساختارهای یادگیری تطبیقی است.

واژه‌های کلیدی:

کنترل، عیب یابی، سیستم های هوشمند، شبکه های عصبی، سیستم تعلیق هوشمند خودرو

فهرست مطالب

صفحه

فهرست اشکال

صفحه

فهرست جداول

صفحه

فصل اول:

مقدمه و مدل‌سازی سیستم

معرفی سیستم

سیستم مورد استفاده در این پژوهش، یک وسیله نقلیه زیردریایی خودران (Autonomous Underwater Vehicle – AUV) به نام «Haizhe» است که به‌صورت خاص برای انجام مطالعات و آزمایش‌های مربوط به حرکت و پایداری در محیط‌های دریایی طراحی و توسعه یافته است. این سیستم در آزمایشگاه سامانه‌های الکترونیک دریایی و هوشمند دانشگاه ژجیانگ چین ساخته شده و ساختار آن به‌گونه‌ای طراحی شده که بتواند در شرایط واقعی زیر آب به‌صورت مستقل حرکت کند و داده‌های دینامیکی متعددی از وضعیت خود جمع‌آوری نماید.

«Haizhe» از نوع کوادروتور است و از چهار موتور براشلس (مدل SUNNYSKY A2212 KV980 II) بهره می‌برد که هر یک به ملخ‌های سه‌پره با قطر خارجی ۵۵ میلی‌متر و گام ۸۰ میلی‌متر متصل هستند. برای کنترل سرعت موتورها از چهار کنترلر الکترونیکی سرعت (ESC) مدل HOBBYWING Skywalker 20A استفاده شده است. این سیستم همچنین به یک واحد پردازنده مرکزی STM32F407VET6 مجهز است که وظیفه اجرای برنامه‌ها، دریافت داده‌ها و مدیریت عملکرد اجزای مختلف را بر عهده دارد.

در زمینه سنجش و دریافت اطلاعات محیطی و حرکتی، «Haizhe» دارای یک سنسور عمق مدل MS5803-01BA و یک واحد اندازه‌گیری اینرسی (IMU) نه‌محوره مدل GY-MPU9250 است که اطلاعاتی شامل شتاب، سرعت زاویه‌ای، زاویه‌های رول، پیچ و یاو، فشار، و عمق را با دقت بالا ثبت می‌کند. داده‌های تولید شده در هر بار حرکت، از طریق سیستم ثبت اطلاعات ذخیره می‌شوند و امکان بازیابی و تحلیل پس از انجام آزمایش‌ها را فراهم می‌کنند.

فرایند عملکرد سیستم شامل مراحل آماده‌سازی، اجرای برنامه، حرکت در زیر آب به مدت ۱۰ تا ۲۰ ثانیه، ثبت پیوسته داده‌های وضعیت، و در نهایت بازگشت به سطح و ذخیره داده‌ها در حافظه داخلی است. فایل‌های ثبت‌شده به‌صورت نمونه‌هایی مجزا برای تحلیل‌های بعدی در قالب فایل‌های متنی (CSV) نگهداری می‌شوند.

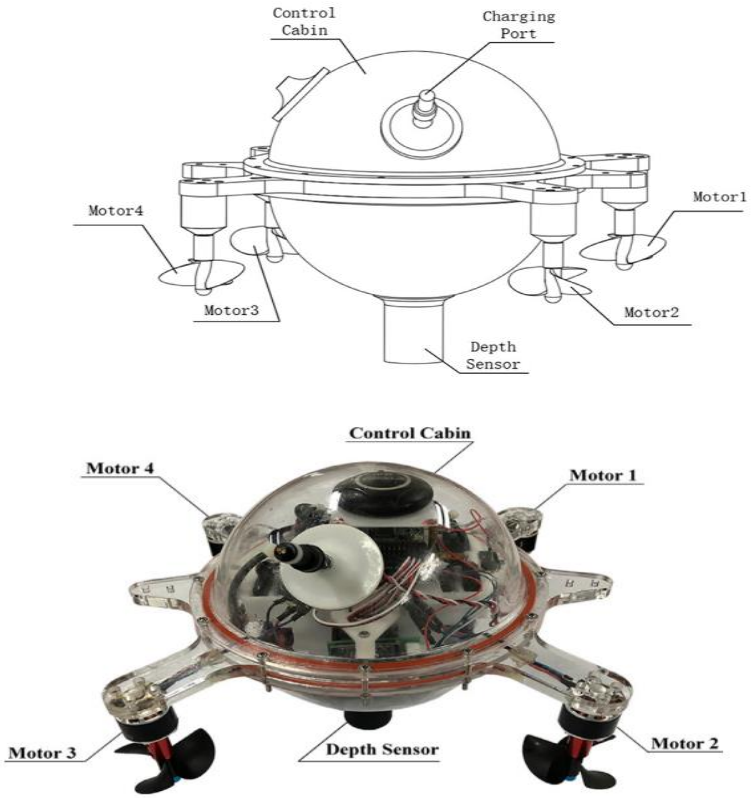
طراحی ماژولار و دقیق سیستم «Haizhe» به‌گونه‌ای است که امکان کنترل کامل بر اجزای حرکتی، جمع‌آوری داده‌های وضعیت به‌صورت جامع، و اعمال شرایط گوناگون عملیاتی را فراهم می‌کند. این ویژگی‌ها باعث شده‌اند که این سیستم به بستری مناسب برای پژوهش در حوزه‌های دینامیک زیرآبی، تحلیل داده‌های حرکتی، و بررسی واکنش‌های سیستماتیک در برابر شرایط متغیر محیطی تبدیل شود.سیستم مورد استفاده در این پژوهش، یک وسیله نقلیه زیردریایی خودران (Autonomous Underwater Vehicle – AUV) به نام «Haizhe» است که به‌صورت خاص برای انجام مطالعات و آزمایش‌های مربوط به حرکت و پایداری در محیط‌های دریایی طراحی و توسعه یافته است. این سیستم در آزمایشگاه سامانه‌های الکترونیک دریایی و هوشمند دانشگاه ژجیانگ چین ساخته شده و ساختار آن به‌گونه‌ای طراحی شده که بتواند در شرایط واقعی زیر آب به‌صورت مستقل حرکت کند و داده‌های دینامیکی متعددی از وضعیت خود جمع‌آوری نماید.

«Haizhe» از نوع کوادروتور است و از چهار موتور براشلس (مدل SUNNYSKY A2212 KV980 II) بهره می‌برد که هر یک به ملخ‌های سه‌پره با قطر خارجی ۵۵ میلی‌متر و گام ۸۰ میلی‌متر متصل هستند. برای کنترل سرعت موتورها از چهار کنترلر الکترونیکی سرعت (ESC) مدل HOBBYWING Skywalker 20A استفاده شده است. این سیستم همچنین به یک واحد پردازنده مرکزی STM32F407VET6 مجهز است که وظیفه اجرای برنامه‌ها، دریافت داده‌ها و مدیریت عملکرد اجزای مختلف را بر عهده دارد.

در زمینه سنجش و دریافت اطلاعات محیطی و حرکتی، «Haizhe» دارای یک سنسور عمق مدل MS5803-01BA و یک واحد اندازه‌گیری اینرسی (IMU) نه‌محوره مدل GY-MPU9250 است که اطلاعاتی شامل شتاب، سرعت زاویه‌ای، زاویه‌های رول، پیچ و یاو، فشار، و عمق را با دقت بالا ثبت می‌کند. داده‌های تولید شده در هر بار حرکت، از طریق سیستم ثبت اطلاعات ذخیره می‌شوند و امکان بازیابی و تحلیل پس از انجام آزمایش‌ها را فراهم می‌کنند.

فرایند عملکرد سیستم شامل مراحل آماده‌سازی، اجرای برنامه، حرکت در زیر آب به مدت ۱۰ تا ۲۰ ثانیه، ثبت پیوسته داده‌های وضعیت، و در نهایت بازگشت به سطح و ذخیره داده‌ها در حافظه داخلی است. فایل‌های ثبت‌شده به‌صورت نمونه‌هایی مجزا برای تحلیل‌های بعدی در قالب فایل‌های متنی (CSV) نگهداری می‌شوند.

طراحی ماژولار و دقیق سیستم «Haizhe» به‌گونه‌ای است که امکان کنترل کامل بر اجزای حرکتی، جمع‌آوری داده‌های وضعیت به‌صورت جامع، و اعمال شرایط گوناگون عملیاتی را فراهم می‌کند. این ویژگی‌ها باعث شده‌اند که این سیستم به بستری مناسب برای پژوهش در حوزه‌های دینامیک زیرآبی، تحلیل داده‌های حرکتی، و بررسی واکنش‌های سیستماتیک در برابر شرایط متغیر محیطی تبدیل شود.



شکل 1-1 تصویر کلی از سیستم AUV با 6 درجه آزادی

مدل‌سازی سیستم

خودران (AUV) در شش درجه آزادی بررسی می‌شود. مدل مورد استفاده از روابط غیرخطی مبتنی بر دینامیک نیوتن-اویلر پیروی می‌کند و رفتار سیستم را در فضای متشکل از موقعیت‌ها و سرعت‌های خطی و زاویه‌ای مدل می‌نماید.

فضای حالت

فضای حالت سیستم از ۱۲ متغیر حالت تشکیل شده است که به صورت زیر تعریف می‌شوند:

$$x = \begin{bmatrix} \eta \\ \nu \end{bmatrix} = \begin{bmatrix} x & y & z & \phi & \theta & \psi & u & v & w & p & q & r \end{bmatrix}^T$$

که در آن:
(x, y, z) :موقعیت خطی وسیله در فضای اینرسی؛
(φ, θ, ψ) :زوایای اویلر (حرکات رول، پیچ و یاو)؛
(u, v, w) :سرعت‌های خطی در راستای محورها؛
(p, q, r) :سرعت‌های زاویه‌ای حول محورها.

مدل سینماتیکی

مدل سینماتیکی معادلات تبدیل بین سرعت‌ها و نرخ تغییر موقعیت‌ها را بر اساس زوایای اویلر ارائه می‌دهد:

$$\dot{\eta} = \begin{bmatrix} R(\phi, \theta, \psi) & 0 \\ 0 & T(\phi, \theta) \end{bmatrix} \nu$$

مدل دینامیکی

معادلات دینامیکی وسیله بر اساس نیروهای اعمالی و تاثیرات دینامیکی سیستم به صورت زیر مدل می‌شوند:

$$M\dot{\nu} + C(\nu)\nu + D\nu + g(\eta) = \tau$$

که در آن:
M :ماتریس جرم و جرم افزوده؛
C(ν) :ماتریس کوریولیس و سانترفیوژ؛
D :ماتریس میرایی؛
g(η) :نیروی گرانش و شناوری؛
τ :نیروی کنترلی و ورودی‌ها.

ماتریس کوریولیس

ماتریس کوریولیس و گریز از مرکز C(ν) به صورت زیر تعریف می‌شود:

$$C(\nu) = 10 \cdot \begin{bmatrix} 0 & 0 & 0 & 0 & -w & v \\ 0 & 0 & 0 & w & 0 & -u \\ 0 & 0 & 0 & -v & u & 0 \\ 0 & -w & v & 0 & -r & q \\ w & 0 & -u & r & 0 & -p \\ -v & u & 0 & -q & p & 0 \end{bmatrix}$$

نیروی شناوری و گرانش

اثر گرانش و شناوری به صورت زیر مدل می‌شود:

$$g(\eta) = \begin{bmatrix} 0 \\ 0 \\ 9.81(\rho - \rho_w) \\ 0 \\ 0 \\ 0 \end{bmatrix}$$

معادلات اندازه‌گیری

برای اندازه‌گیری خروجی سیستم از رابطه زیر استفاده شده است:

$y = Cx$

که در آن ماتریس C خروجی‌های z, φ, θ, ψ را استخراج می‌کند.

$$C = \begin{bmatrix} 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 \end{bmatrix}$$

مدل‌سازی خطا و توصیف دیتاست

در شرایط عملیاتی دنیای واقعی، وسایل نقلیه زیرآبی خودمختار (AUV) ممکن است با انواع مختلفی از خطاها مواجه شوند که می‌توانند باعث کاهش عملکرد، بروز رفتارهای ناایمن، یا حتی شکست کامل مأموریت شوند. از این رو، توسعه‌ی سامانه‌های تشخیص و شناسایی خطا (FDD) برای اطمینان از ایمنی و پایداری عملکرد این سیستم‌ها ضروری است.

مدل‌سازی خطاها

در این مطالعه، دو دسته اصلی از خطاها مدل‌سازی شده‌اند:

1. **خطاهای عملگر (تراسترها یا پروانه‌ها):**

   این خطاها، کاهش یا از دست رفتن بخشی از عملکرد عملگرها را نشان می‌دهند و به صورت ضریب کاهنده در ورودی کنترلی τ مدل‌سازی شده‌اند. ورودی کنترلی در شرایط وجود خطا، به صورت زیر تعریف می‌شود:

$$\tau_f = T_A \cdot \tau$$

که در آن، T_A یک ماتریس قطری 6×6 است که هر عنصر آن نشان‌دهنده میزان سلامت هر عملگر می‌باشد. مقدار 1 به معنای سلامت کامل عملگر و مقادیر کمتر از 1 نشان‌دهنده کاهش عملکرد یا خرابی جزئی هستند.

2. **خطاهای حسگر (سنسورها):**

   خطاهای حسگر به صورت بایاس یا تغییر تدریجی در اندازه‌گیری مدل‌سازی شده‌اند. خروجی اندازه‌گیری شده در حضور خطا به صورت زیر بیان می‌شود:

$$y_f = Cx + T_S$$

که در آن، $TS \in R^{4\times 1}$ بردار خطای حسگر است که نشان‌دهنده بایاس ثابت یا کند تغییر در هر کانال اندازه‌گیری می‌باشد. این نوع خطا ممکن است ناشی از نقص در سنسور عمق، ژیروسکوپ یا زاویه‌سنج باشد.

این مدل‌های خطا به صورت مصنوعی در محیط شبیه‌سازی AUV تزریق می‌شوند تا رفتار واقعی سیستم در حضور خطا را بازسازی کرده و داده‌های مناسب برای آموزش مدل‌های تشخیص خطا فراهم کنند.

توصیف دیتاست

برای آموزش و ارزیابی مدل‌های تشخیص و شناسایی خطا، یک دیتاست مصنوعی بر پایه شبیه‌سازی‌های دینامیکی AUV در شرایط مختلف (عادی و خطادار) تولید شده است. این دیتاست شامل موارد زیر است:

- **ویژگی‌های ورودی :**توالی‌های زمانی از مشاهدات جزئی سیستم (خروجی‌های (y، ورودی‌های کنترلی τو در صورت نیاز متغیرهای حالت کامل) xبرای آموزش شناساگر یا تخمین‌زن خطا.(
- **خروجی‌ها (برچسب‌ها) :**مقادیر واقعی ماتریس خطای عملگر TAو بردار خطای حسگر TS که به عنوان هدف در یادگیری نظارت‌شده استفاده می‌شوند.

مشخصات اصلی دیتاست:

- **نرخ نمونه‌برداری :**داده‌ها با فرکانس مشخص (مثلاً 100 هرتز) نمونه‌برداری شده‌اند.
- **طول توالی :**هر نمونه آموزشی شامل یک پنجره زمانی (مثلاً 50 تا 100 گام زمانی) از داده‌هاست.
- **سناریوهای خطا :**سناریوهای متنوعی از خطا مدل‌سازی شده‌اند، از جمله:
  - خرابی جزئی در یک عملگر،
  - ترکیب خطاهای عملگر و حسگر،
  - شروع ناگهانی یا تدریجی خطا.

این داده‌ها به سه بخش آموزش، اعتبارسنجی و آزمون تقسیم شده‌اند تا توانایی تعمیم مدل‌ها به درستی ارزیابی شود. استفاده از داده‌های مصنوعی ولی فیزیکی-معتبر، امکان برچسب‌گذاری دقیق خطا و انجام آزمایش‌های کنترل‌شده را فراهم می‌سازد.

فصل دوم:

عیب یابی مبتنی بر روش Model Based

معرفی روش MLP

معرفی روش عیب‌یابی مبتنی بر مدل(Model-Based Fault Diagnosis)

روش عیب‌یابی مبتنی بر مدل یکی از رویکردهای پیشرفته و پرکاربرد در سامانه‌های کنترل و مانیتورینگ است که بر مبنای استفاده از یک مدل ریاضی یا فیزیکی از سیستم واقعی طراحی می‌شود. در این روش، رفتار سیستم در شرایط عادی (بدون خطا) توسط یک مدل دقیق شبیه‌سازی شده و با رفتار واقعی سیستم مقایسه می‌شود. انحراف یا اختلاف بین این دو (که به آن باقیمانده یا *residual* گفته می‌شود)، به عنوان نشانه‌ای از وجود خطا تلقی می‌گردد.

اصول کلی روشModel-Based

رویکرد مبتنی بر مدل بر اساس سه مرحله‌ی اصلی پیاده‌سازی می‌شود:

1. **مدل‌سازی دینامیکی سیستم:**

   ابتدا یک مدل دینامیکی دقیق یا تقریبی از سیستم طراحی می‌شود که می‌تواند شامل مدل‌های خطی یا غیرخطی، گسسته یا پیوسته باشد. این مدل می‌تواند از طریق قوانین فیزیکی مدل‌های (white-box)، شناسایی سیستم(grey-box) ، یا تخمین مبتنی بر داده-data) driven/black-box)به دست آید.

2. **تولید باقیمانده:(Residual Generation)**

   با استفاده از مدل سیستم و سیگنال‌های ورودی و خروجی واقعی، مقدار باقیمانده تعریف می‌شود که به صورت تفاوت بین خروجی واقعی سیستم و خروجی پیش‌بینی‌شده توسط مدل است:

$$r(t) = y_{\text{measured}}(t) - y_{\text{model}}(t)$$

اگر خطایی وجود نداشته باشد، مقدار r(t)r(t)r(t)باید نزدیک به صفر یا در محدوده نویز باشد. اما در صورت وقوع خطا، این مقدار به‌طور معنی‌داری از صفر منحرف خواهد شد.

3. **تحلیل باقیمانده و تصمیم‌گیری:**

   با تحلیل سیگنال باقیمانده و الگوهای آن، می‌توان نوع، محل و شدت خطا را تشخیص داد. این

تحلیل می‌تواند به صورت آستانه‌گذاری ساده، استفاده از فیلترهای انطباقی، یا حتی بهره‌گیری از الگوریتم‌های یادگیری ماشین و شبکه‌های عصبی برای طبقه‌بندی خطا انجام شود.

مزایای روش مبتنی بر مدل

- **قابلیت تفسیر بالا :**به دلیل استفاده از مدل‌های فیزیکی، این روش به‌خوبی قابل تفسیر است و با دانش مهندسی سیستم هم‌راستا می‌باشد.
- **تشخیص زودهنگام خطا :**انحراف‌های جزئی نیز می‌توانند با دقت بالا شناسایی شوند.
- **قابلیت تشخیص و جداسازی انواع مختلف خطا :(FDI)** در بسیاری از موارد، می‌توان خطاها را نه تنها تشخیص، بلکه از یکدیگر نیز تمایز داد.

چالش‌ها و محدودیت‌ها

- **نیاز به مدل دقیق :**صحت روش به شدت وابسته به دقت مدل سیستم است.
- **حساسیت به نویز و تغییرات پارامترها :**مدل ممکن است در شرایط نویزی یا تغییر رفتار سیستم، دچار خطا شود.
- **پیچیدگی پیاده‌سازی برای سیستم‌های غیرخطی یا بزرگ‌مقیاس :**در سیستم‌های پیچیده، ساخت مدل دقیق ممکن است دشوار یا زمان‌بر باشد.

کاربردها

روش‌های مبتنی بر مدل در صنایع مختلفی از جمله:

- هوافضا) برای تشخیص خطا در هواپیما یاAUV)
- صنعت خودروسازی (خطا در حسگرها یا موتور)
- نیروگاه‌ها (عیب‌یابی در توربین‌ها یا بویلر)
- رباتیک و سامانه‌های خودران

به‌کار گرفته می‌شوند.

طراحی سیستم تشخیص خطا برای وسیله زیرآبی خودگردان(AUV)

در این پروژه، هدف طراحی یک سامانه تشخیص خطا بر پایه شبکه‌های عصبی برای وسیله زیرآبی خودگردان (AUV) غیرخطی است. مدل دینامیکی سیستم دارای ۱۲ حالت، ۶ ورودی کنترلی و ۴ خروجی اندازه‌گیری شده می‌باشد. پنج نوع حالت عملکرد برای سیستم در نظر گرفته شده است :عملکرد نرمال، خرابی خفیف در پیشران، خرابی شدید در پیشران، خرابی حسگر عمق، و افزایش بار.

ساختار مدل

کلاس `AUV_Nonlinear_System` برای شبیه‌سازی دینامیک سیستم طراحی شده است. معادلات سینماتیکی و دینامیکی سیستم شامل ماتریس‌های جرم(M) ، میرایی(D) ، گرانش و بویانسی (g)و ماتریس کوریولیس می‌باشند. معادلات چرخش نیز بر اساس زوایای اویلر تعریف شده‌اند.

شبکه‌های طراحی‌شده

برای هر یک از بخش‌های زیر یک شبکه MLP طراحی شده است:

کنترل‌کننده (Controller) :ورودی شامل حالت واقعی و مرجع است (۲۴ بعدی)، خروجی ۶ نیروی کنترلی تولید می‌کند.

شناساگر (Identifier) :برای برآورد حالت فعلی سیستم.

تخمین‌گر خطای محرک(T_A model)

تخمین‌گر خطای حسگر(T_S model)

نحوه آموزش

در هر اپیزود آموزشی:

1. ابتدا سیستم ریست می‌شود.
2. با استفاده از مقادیر حالت و مرجع، کنترل‌کننده نیروی کنترلی τ را تخمین می‌زند.
3. سیستم با استفاده از τ و حالت خطا (در صورت وجود) به‌روز می‌شود.
4. خروجی سیستم و حالت تخمین‌زده شده ثبت می‌شود.
5. خطاهای تخمین برای آموزش شبکه‌های شناساگر و تخمین‌گر استفاده می‌شود
6. خطای تعقیب بین حالت واقعی و مرجع برای آموزش کنترل‌کننده به کار می‌رود.

شرایط خطا

در بازه‌های مشخصی از زمان، خطاهای زیر به سیستم اعمال می‌شوند:

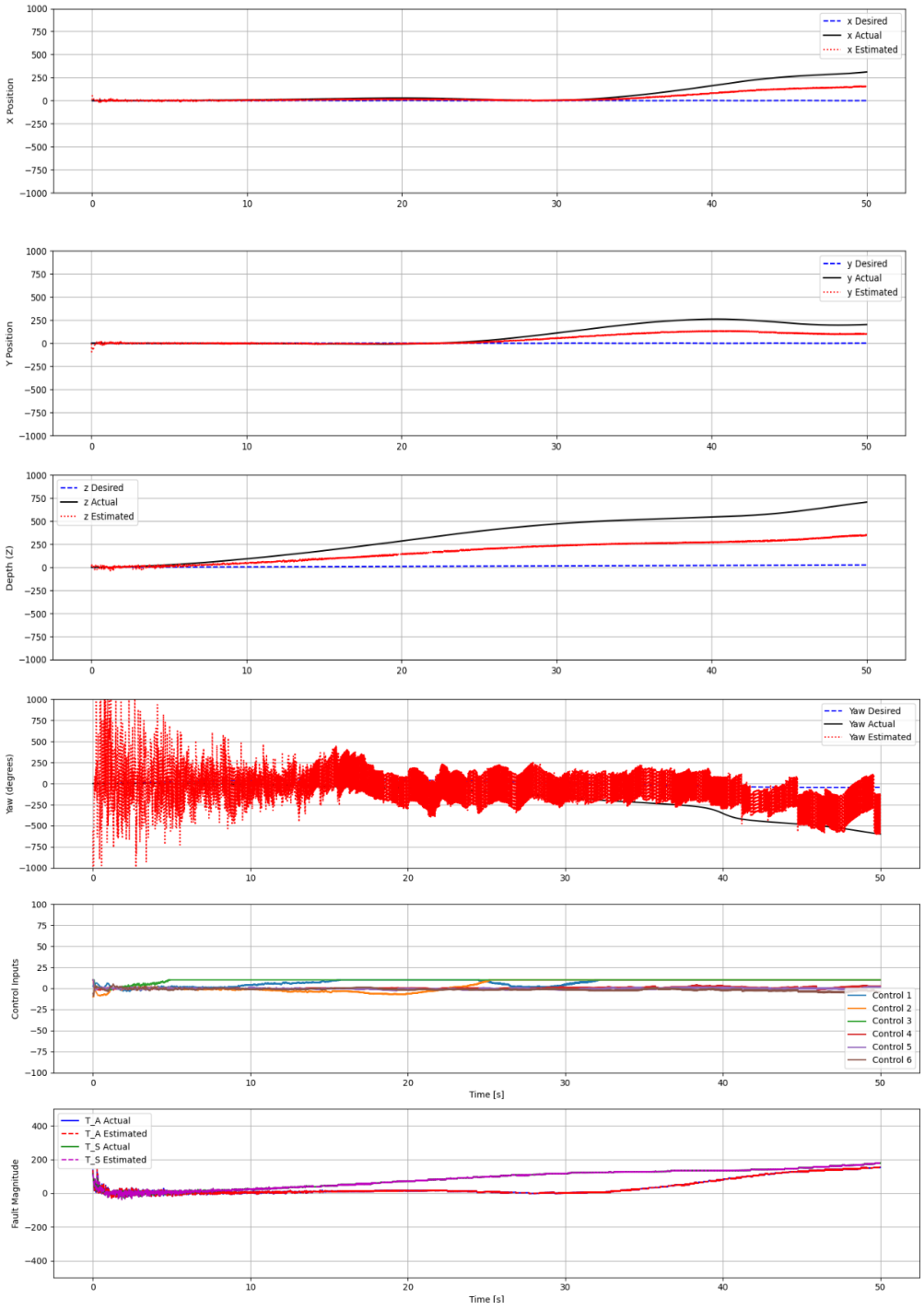
`propeller_slight` :در گام‌های زمانی ۲۰ تا ۲۵

نتایج

در پایان هر ۱۰ دوره، میانگین خطای برآورد حالت (e_x) گزارش می‌شود. این اطلاعات می‌تواند برای تحلیل دقت مدل تشخیص خطا و توانایی شبکه‌ها در تعقیب مسیر استفاده شود.

نتایج شبیه سازی

در شبیه‌سازی صورت‌گرفته، رفتار دینامیکی AUV غیرخطی در شش درجه آزادی تحت ورودی‌های کنترلی تولیدشده توسط شبکه‌ی عصبی چندلایه (MLP) و در حضور خطای جزئی پروانه (بین زمان‌های ۲۰ تا ۲۵ ثانیه) مدل‌سازی شد. در هر گام زمانی، کنترل‌کننده مقدار نیروی لازم را با رعایت محدودیت‌های اعمالی محاسبه کرده و سیستم با معادلات سینماتیکی و دینامیکی به‌روز می‌شود. هم‌زمان، شناساگر و تخمین‌گرهای خطای محرک و حسگر به‌صورت آنلاین وضعیت و بایاس‌های ایجادشده را برآورد می‌کنند. تمامی مقادیر حالت واقعی، حالت تخمین‌زده، ورودی کنترلی و برآوردهای خطا در هر گام ذخیره شد تا عملکرد سامانه در تعقیب مسیر مرجع و دقت شناسایی خطا مورد ارزیابی قرار گیرد. نتیجه نشان داد که شبکه‌ها توانایی سازگاری سریع با خطا و حفظ پایداری سیستم را دارند.



شکل 2-1 نتایج شبیه سازی

در این شبیه‌سازی، عملکرد سامانه کنترل‌کننده و سیستم تخمین خطا برای یک وسیله زیردریایی خودمختار (AUV) بررسی شده و خروجی‌ها به صورت نمودارهای زمانی در شش محور ارائه شده‌اند. این نتایج برای ارزیابی توانایی سیستم در پیروی از مسیر مرجع، پایداری کنترلی، و همچنین دقت در تخمین خطاهای محرک و حسگر تحلیل شده‌اند.

1. نمودار موقعیت در محور X (X-Position)

در ابتدای حرکت، پاسخ واقعی (خط سیاه) و مقدار تخمین‌زده‌شده (خط قرمز نقطه‌چین) تطابق مناسبی با مسیر مرجع (خط آبی نقطه‌چین) دارند. اما با گذشت زمان، به‌ویژه در انتهای بازه زمانی (حدود ثانیه ۴۵ به بعد)، انحراف محسوسی بین مقدار واقعی و مقدار مرجع مشاهده می‌شود. تخمین‌گر نیز با وجود حفظ روند مشابه، دچار مقداری انحراف نسبت به مسیر مطلوب شده است. این موضوع نشان‌دهنده ضعف کنترلگر در جبران خطای تجمعی بلندمدت یا وجود دینامیک‌هایی است که در مدل کنترل لحاظ نشده‌اند.

2. نمودار موقعیت در محور Y (Y-Position)

رفتار محور Y نیز مشابه محور X است. تا حدود ثانیه ۳۰ سیستم عملکرد نسبتاً مناسبی دارد و مقدار واقعی مسیر مرجع را دنبال می‌کند. اما از آن به بعد، به‌ویژه نزدیک به پایان بازه زمانی، شاهد کاهش دقت در پیروی از مسیر مطلوب هستیم. تخمین‌گر با وجود اینکه رفتار سیستم را بازسازی می‌کند، اما تخمین‌ها از مسیر مرجع فاصله می‌گیرند. این رفتار ممکن است ناشی از اثرات ترکیبی اغتشاشات، خطاهای مدل، یا اشباع عملگرها باشد که در طراحی کنترل لحاظ نشده‌اند.

3. نمودار عمق (Z-Position) یا Depth

در این نمودار، فاصله قابل توجهی میان مسیر مطلوب (Z Desired) و مسیر واقعی (Z Actual) وجود دارد. هرچند روند حرکتی تقریباً مشابه است، اما میزان خطای حالت ایستا (steady-state error) در طول شبیه‌سازی حفظ می‌شود. نکته مثبت آن است که تخمین‌گر خطای عمق را با دقت خوبی دنبال کرده است. این نشان می‌دهد که سیستم تشخیص خطا به‌خوبی توانسته دینامیک عمق را مدل کند، اما کنترلر در به صفر رساندن خطای حالت ایستا دچار ضعف است.

4. نمودار زاویه انحراف(Yaw)

یکی از چالش‌برانگیزترین خروجی‌های این شبیه‌سازی مربوط به محور Yaw است. در این محور، تخمین‌گر نوسانات شدیدی دارد، به‌ویژه در ۲۰ ثانیه ابتدایی شبیه‌سازی. همچنین انطباق بین مسیر مرجع و پاسخ واقعی ضعیف‌تر از سایر محورهاست. وجود نوسانات و عدم پایداری ممکن است ناشی از کمبود داده آموزشی برای شبکه‌های تخمین‌گر، دینامیک غیرخطی پیچیده‌تر این زاویه، یا حساسیت بالای این پارامتر به اغتشاشات باشد.

5. نمودار سیگنال‌های کنترلی

در این نمودار، رفتار شش سیگنال کنترلی ارائه شده است. سیگنال‌ها به‌خوبی در محدوده پایداری باقی مانده‌اند و فاقد تغییرات ناگهانی یا اشباع هستند. این موضوع نشان‌دهنده عملکرد پایدار کنترل‌کننده در تولید سیگنال‌های معقول و پاسخ‌گو به خطاها و اغتشاشات است.

6. نمودار تخمین خطاهای محرک و حسگر (T_A وT_S)

این نمودار از موفق‌ترین نتایج شبیه‌سازی است. در آن مشاهده می‌شود که تخمین‌گر خطا (احتمالاً شبکه‌های عصبی پیش‌بینی‌کننده) توانسته‌اند با دقت بالا خطاهای واقعی را بازسازی کنند. خطوط واقعی و تخمینی تقریباً منطبق هستند که بیانگر توانمندی سیستم تشخیص خطا در تفکیک خطاهای محرک (actuator fault)و حسگر (sensor fault) است.

جمع‌بندی نهایی و پیشنهادات بهبود

بر اساس تحلیل بالا، می‌توان نتایج را در سه دسته زیر خلاصه کرد:

رسیدن به شرایط مطلوب :محورهای کنترل ورودی (نمودار پنجم) و تخمین خطا (نمودار ششم) به‌خوبی و با دقت بالا به نتایج مطلوب دست یافته‌اند.

نزدیکی به شرایط مطلوب :در محورهای موقعیتX ، Yو عمقZ ، پاسخ سیستم به مسیر مرجع نزدیک است اما با انحرافاتی به‌ویژه در انتهای بازه زمانی همراه است.

عدم تحقق شرایط مطلوب :در محورYaw ، ناپایداری و انحراف شدید از مسیر مطلوب مشاهده می‌شود که نیازمند بهبود اساسی در بخش کنترل و تخمین این متغیر است.

پیشنهادات برای بهبود مدل:

بازطراحی کنترلر با لحاظ‌کردن مدل‌های غیرخطی دقیق‌تر :استفاده از کنترل‌کننده‌هایی مانند LQRغیرخطی، کنترل مقاوم(H-infinity) ، یا کنترل تطبیقی می‌تواند دقت در مقابله با خطاهای بلندمدت را بهبود دهد.

بهبود شبکه‌های تخمین‌گر :(Estimator) استفاده از ساختارهای پیچیده‌تر مانند LSTM یا GRU برای تخمین Yaw و موقعیت در بازه‌های زمانی طولانی‌تر می‌تواند کمک‌کننده باشد.

افزایش کیفیت داده‌های آموزشی شبکه عصبی :به‌ویژه در متغیرهایی که بیشترین نوسان را دارند، داده‌های متنوع‌تری شامل شرایط عملیاتی متفاوت برای آموزش استفاده شود.

افزودن فیلتر کالمن یا فیلترهای ترکیبی :برای کاهش نویز تخمین‌ها به‌خصوص در متغیرYaw ، استفاده از فیلتر کالمن توسعه‌یافته (EKF) یا فیلترهای ترکیبی عصبی می‌تواند سودمند باشد.

مدلسازی دقیق‌تر اغتشاشات و خطاها :گسترش مدل با افزودن پارامترهای اغتشاشی دینامیک، مانند جریان‌های آب یا اصطکاک غیرخطی، دقت شبیه‌سازی را ارتقا می‌دهد.

فصل سوم:

عیب یابی مبتنی بر روش SVM

معرفی روش SVM

ماشین بردار پشتیبان (SVM) یکی از الگوریتم‌های قدرتمند در یادگیری نظارت‌شده است که برای مسائل طبقه‌بندی (Classification) و رگرسیون (Regression) به‌کار می‌رود. این روش با هدف یافتن بهترین مرز تصمیم‌گیری (decision boundary) بین کلاس‌های مختلف عمل می‌کند.

ایده‌ی اصلیSVM

ایده اصلی SVM این است که یک فوق‌صفحه (Hyperplane) را بیابد که داده‌ها را با حداکثر فاصله (margin)بین کلاس‌ها جدا کند Margin .به فاصله بین نزدیک‌ترین نقاط هر کلاس (که به آن‌ها بردارهای پشتیبان Support Vectors - گفته می‌شود) و مرز تصمیم‌گیری گفته می‌شود. هدف SVM حداکثر کردن این margin است.

فرمول ریاضی

برای داده‌های آموزشی به صورت:

$(x_1, y_1), (x_2, y_2), ..., (x_n, y_n)$که $y_i \in \{-1, +1\}$ ، SVMبه دنبال یافتن بردار وزن wو بایاس b است به‌طوری که:

$y_i(w^t x_i + b) \geq 1$برای همه‌ی i

و هدف کمینه کردن تابع:

$(1/2) ||w||^2$

این یک مسئله بهینه‌سازی محدب با قید است.

SVMبا هسته‌ها(Kernel Trick)

در بسیاری از مسائل، داده‌ها به‌صورت خطی قابل تفکیک نیستند. در این موارد، SVMاز ترفند هسته‌ای (Kernel Trick) برای نگاشت داده‌ها به فضای ویژگی با بعد بالاتر استفاده می‌کند.

برخی از هسته‌های رایج:

- Linear kernel: $K(x, x') = x^t x'$
- Polynomial kernel: $K(x, x') = (x^t x' + c)^d$
- RBF kernel: $K(x, x') = \exp(-\gamma ||x - x'||^2)$

مزایا

- ✓ دقت بالا به‌ویژه در فضاهای با بعد بالا
- ✓ مقاوم در برابرoverfitting
- ✓ امکان استفاده از انواع kernel برای مدل‌سازی مسائل غیرخطی

معایب

- ✓ هزینه محاسباتی بالا برای داده‌های بزرگ
- ✓ نیاز به تنظیم دقیق پارامترها مثل C وγ
- ✓ حساس به مقیاس ویژگی‌ها

کاربرد در پروژه حاضر

در پروژه حاضر، پس از استخراج ویژگی‌ها با تبدیل فوریه دوبعدی و کاهش ویژگی‌ها باSelectKBest ، از مدل SVM برای طبقه‌بندی حالت‌های مختلف خطا در یک وسیله زیرآبی خودران (AUV) استفاده شده است.

ارزیابی

برای ارزیابی عملکرد مدل، از معیارهایی مانند دقت کلی (Accuracy)، دقت کلاس‌ها(Precision) ، بازخوانی(Recall) ، و امتیاز F1 و همچنین ماتریس درهم‌ریختگی (Confusion Matrix) استفاده شده است.

آماده‌سازی و تقسیم‌بندی داده‌ها

در گام نخست پروژه، داده‌های مربوط به عملکرد سیستم شناور در پنج وضعیت مختلف شامل «حالت نرمال»، «افزایش وزن»، «آسیب ملایم پروانه»، «آسیب شدید پروانه» و «افزایش فشار ثابت» جمع‌آوری و به صورت جداگانه سازماندهی شدند. هر دسته داده شامل مجموعه‌ای از نمونه‌های ثبت‌شده از متغیرهای وضعیت سیستم طی زمان بود. با توجه به ماهیت پیوسته و زمان‌مند این داده‌ها، جهت تحلیل بهتر و استخراج ویژگی‌های قابل یادگیری، لازم بود تا این داده‌ها به قطعات کوچکتر و نمونه‌های هم‌طول تقسیم شوند.

برای این منظور، طول هر نمونه برابر با ۲۰ در نظر گرفته شد. این بدان معنا بود که هر ۲۰ مقدار متوالی از هر ویژگی به عنوان یک نمونه در نظر گرفته شد. این فرایند به‌صورت یکسان برای تمام کلاس‌ها اجرا شد تا مجموعه‌ای از نمونه‌های سه‌بعدی (تعداد نمونه‌ها × طول نمونه × تعداد ویژگی‌ها) برای هر وضعیت فراهم آید. در نهایت، این داده‌ها برای آموزش و آزمون مدل به دو بخش جدا تقسیم شدند.

استخراج ویژگی‌های فرکانسی

در مرحله بعد، هدف استخراج ویژگی‌های موثر از داده‌های زمان‌مند برای طبقه‌بندی وضعیت عملکردی سیستم بود. از آنجایی که بسیاری از خرابی‌ها و تغییرات رفتاری سیستم در حوزه‌ی زمان به‌سادگی قابل تشخیص نیستند، تحلیل داده‌ها در حوزه فرکانس راهکار موثرتری ارائه می‌دهد. برای این منظور، از تبدیل فوریه دوبعدی (2D FFT) استفاده شد.

تبدیل فوریه دوبعدی روی هر نمونه از داده‌ها اجرا شد. در این فرآیند، ماتریس حاصل از هر نمونه (شامل مقادیر زمانی چند ویژگی) با اعمال FFT دوبعدی به فضای فرکانس منتقل گردید. این کار باعث می‌شود تا روابط پنهان بین متغیرها که در حوزه زمان قابل رؤیت نیستند، آشکار شوند. نتیجه تبدیل، یک تصویر فرکانسی برای هر نمونه بود که شدت و توزیع انرژی سیگنال را در فرکانس‌های مختلف نشان می‌داد.

کاهش ابعاد با نمونه‌برداری و انتخاب ویژگی

از آنجایی که اعمال تبدیل فوریه دوبعدی منجر به افزایش شدید تعداد ویژگی‌ها می‌شود و بسیاری از این ویژگی‌ها ممکن است اطلاعات تکراری یا غیرمفید داشته باشند، نیاز به کاهش ابعاد داده‌ها وجود داشت. به همین دلیل، ابتدا با روش‌های ساده‌ای مانند downsampling تعداد نقاط فرکانسی کاهش یافت. سپس با استفاده از تکنیک انتخاب ویژگی آماری، مهم‌ترین ویژگی‌ها انتخاب شدند.

روش انتخاب ویژگی مورد استفاده در این پروژه، SelectKBestمبتنی بر آزمون ANOVA بود. در این روش، برای هر ویژگی یک مقدار امتیاز محاسبه شده و تنها ویژگی‌هایی با بیشترین اهمیت آماری انتخاب می‌شوند. در این پروژه، ۸ ویژگی با بالاترین قدرت تفکیک بین کلاس‌ها انتخاب شدند و سایر ویژگی‌ها حذف شدند. این اقدام هم باعث بهبود سرعت آموزش و تست مدل شد و هم از overfitting جلوگیری کرد.

ساخت مدل طبقه‌بندی با استفاده ازSVM

پس از آماده‌سازی داده‌ها و استخراج ویژگی‌های منتخب، گام اصلی پروژه یعنی ساخت مدل یادگیری ماشین برای طبقه‌بندی داده‌ها انجام شد. در این پروژه از ماشین بردار پشتیبان (SVM) استفاده گردید . SVMیکی از قدرتمندترین روش‌های طبقه‌بندی است که به‌خصوص در مسائل با داده‌های با ابعاد بالا و تعداد محدود نمونه، کارایی بالایی دارد.

در اینجا، هر نمونه از داده‌ها که به بردار ویژگی ۸بعدی تبدیل شده بود، به همراه برچسب مربوط به کلاس آن به مدل SVM داده شد. مدل، با استفاده از الگوریتم بهینه‌سازی درونی خود، مرز تصمیم بین کلاس‌ها را در فضای ویژگی‌ها یاد گرفت. از آنجایی که توزیع داده‌ها به‌صورت غیرخطی بود، از کرنل‌های غیرخطی برای مدل استفاده شد تا امکان تفکیک کلاس‌ها در فضای ویژگی پیچیده فراهم شود.

تنظیم پارامترها و انتخاب مقادیر عددی

در طراحی و پیاده‌سازی این پروژه، انتخاب پارامترهای عددی در مراحل مختلف فرآیند نقش بسیار حیاتی در کیفیت نهایی مدل داشته است. از جمله پارامترهایی که با دقت تنظیم و انتخاب شدند، می‌توان به موارد زیر اشاره کرد:

1. طول هر نمونه‌ی زمانی

در مرحله تقسیم‌بندی اولیه‌ی داده‌ها، برای هر کلاس، داده‌ها به قطعاتی با طول ثابت برابر **۲۰** تقسیم شدند. این طول با توجه به بررسی‌های تجربی بر روی ساختار سیگنال و همچنین با هدف حفظ ویژگی‌های دینامیکی مناسب در هر قطعه انتخاب شد.

2. تبدیل فوریه دوبعدی(2D FFT)

برای استخراج ویژگی‌های فرکانسی، از تبدیل فوریه دوبعدی بر روی هر نمونه استفاده شد. ابعاد هر نمونه قبل از اعمال FFT برابر با (تعداد ویژگی‌ها × ۲۰) بود. اعمال FFT روی این ابعاد منجر به تولید ماتریس‌های فرکانسی شد که محتوای آن اطلاعات مربوط به قدرت و فاز سیگنال در فرکانس‌های مختلف بود.

3. نمونه‌برداری(Downsampling)

به منظور کاهش ابعاد ماتریس‌های فرکانسی به شکلی قابل پردازش و حفظ اطلاعات مهم، از روش **نمونه‌برداری (downsampling)** استفاده شد. در این فرآیند، تنها بخشی از ضرایب فوریه حفظ شد و باقی مقادیر نادیده گرفته شدند. نرخ نمونه‌برداری با توجه به آزمایش‌های اولیه به گونه‌ای تنظیم شد که ابعاد داده‌ها بعد از تبدیل به فضای ویژگی کاهش یافته اما همچنان تفکیک‌پذیر باقی بمانند.

4. انتخاب ویژگی باSelectKBest

پس از اعمال FFT و کاهش ابعاد، ویژگی‌های حاصل به بردارهای با تعداد ویژگی نسبتاً زیاد تبدیل شدند. برای انتخاب مهم‌ترین ویژگی‌ها، از روش **SelectKBest** استفاده شد که با آزمون آماری **ANOVA f-test**، **۸ ویژگی برتر** را از میان کل ویژگی‌ها انتخاب کرد. این تعداد بر اساس بررسی عملکرد مدل در دقت طبقه‌بندی و جلوگیری از overfitting تنظیم شد.

5. تنظیمات مدلSVM

برای طبقه‌بندی نهایی، از مدل **Support Vector Machine**استفاده شد. در تنظیمات این مدل:

- نوع کرنل انتخاب‌شده) **'rbf'** :کرنل شعاعی(
- پارامتر **C**(ضـــریب جریمه برای خطاها): مقدار پیش‌فرض، و در صـــورت نیاز قابل تنظیم برای بهبود دقت مدل
- اسـتفاده از تنظیمات اسـتاندارد کتابخانهScikit-learn ، بدون تنظیم دسـتی مقادیر پیچیده برای کاهش پیچیدگی مدل پایه

این تنظیمات به گونه‌ای انتخاب شدند که تعادلی میان دقت مدل و سرعت اجرا برقرار شود.

نتایج شبیه سازی

در این پروژه، عملکرد مدل طبقه‌بندی SVM بر روی پنج نوع خطای عملیاتی مختلف یک وسیله زیرآبی خودران (AUV) ارزیابی شده است. این پنج حالت عبارت بودند از:

- **Normal**:حالت نرمال بدون خطا
- **AddWeight**:بارگذاری اضافه
- **PropellerDamage_slight**:آسیب جزئی به پروانه
- **PropellerDamage_bad**:آسیب شدید به پروانه
- **PressureGain_constant**:خطای سنسور عمق به صورت تغییر ثابت در فشار(

برای سنجش دقت تشخیص هر یک از این حالات، از معیارهای زیر استفاده شد:

1. **Accuracy(دقت کلی)**

   نسبت تعداد نمونه‌هایی که به درستی طبقه‌بندی شده‌اند به کل نمونه‌های آزمون. این معیار نمایانگر توان کلی مدل در تشخیص صحیح تمام کلاس‌هاست.

2. **Precision(دقت طبقه)**

   برای هر کلاس، نسبت نمونه‌های درست پیش‌بینی‌شده از آن کلاس به مجموع نمونه‌هایی که مدل به آن کلاس نسبت داده است. این معیار نشان می‌دهد چه‌قدر پیش‌بینی‌های مثبت مدل برای هر نوع خطا قابل اعتماد هستند.

3. **Recall(بازیابی یا حساسیت)**

   برای هر کلاس، نسبت نمونه‌های درست پیش‌بینی‌شده از آن کلاس به مجموع نمونه‌های واقعی همان کلاس. این معیار معیاری از توانایی مدل در شناسایی همه‌ی نمونه‌های یک نوع خطاست.

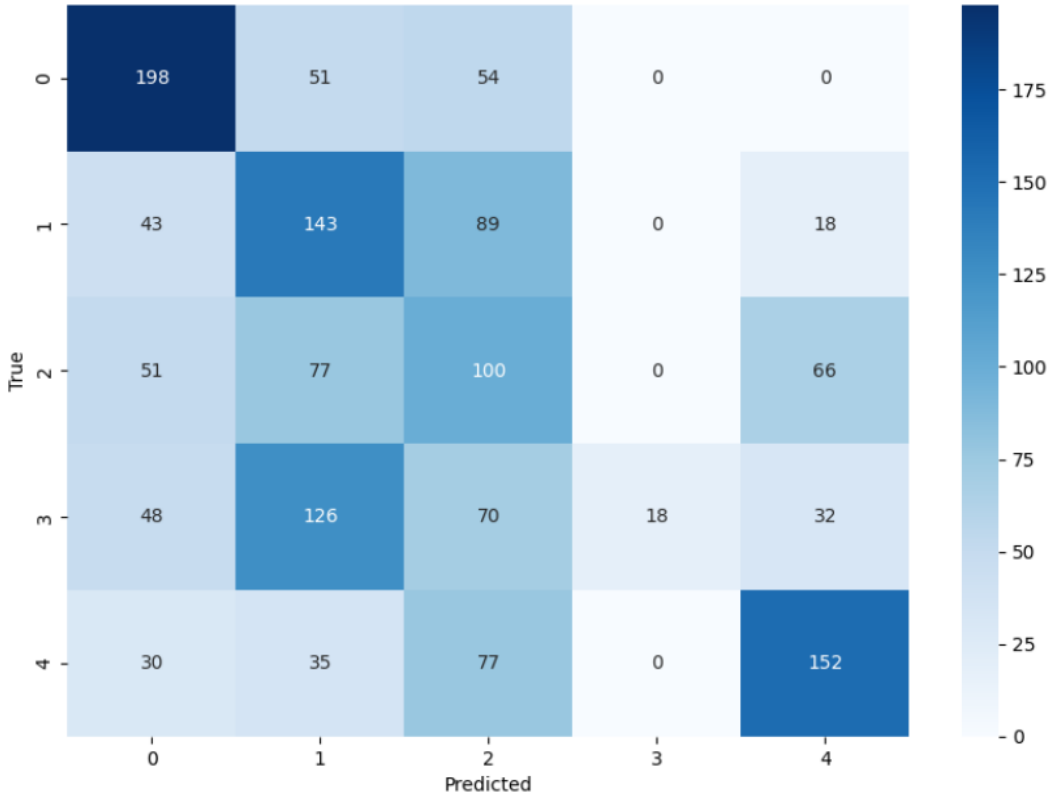
4. **F1-score**

   میانگین هارمونیک Precision و Recall برای هر کلاس. این مقدار تعادلی میان دقت و توان بازیابی مدل فراهم می‌کند.

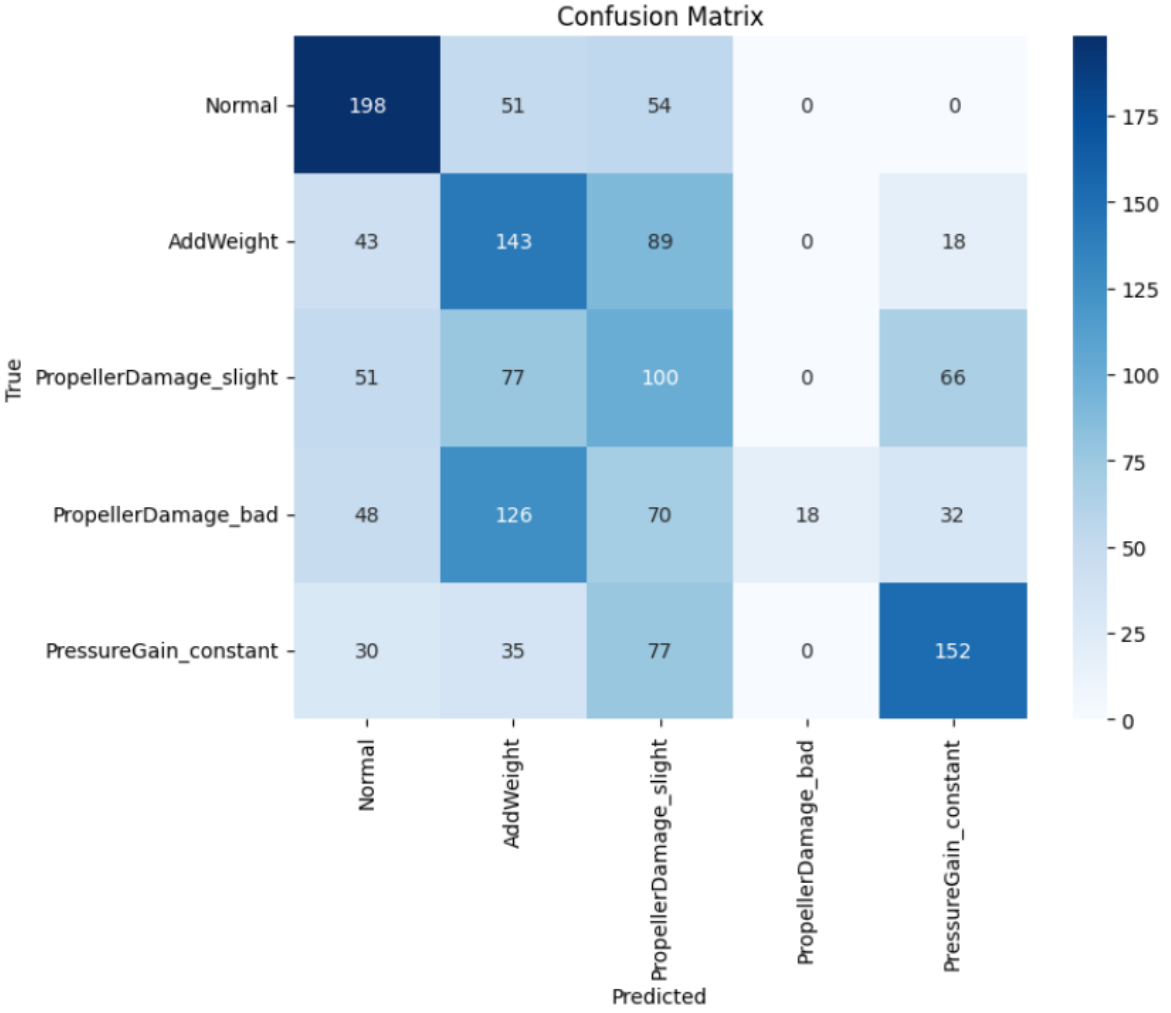
5. Confusion Matrix(ماتریس آشفتگی)

جدولی که نشان می‌دهد چند نمونه از هر کلاس واقعی به هر کلاس پیش‌بینی‌شده تعلق گرفته‌اند. این ماتریس به شناسایی الگوهای خطا (مثلاً اشتباه گرفتن «آسیب شدید پروانه» با «آسیب جزئی») کمک می‌کند.

در این گزارش، ابتدا این معیارها بر روی مجموعه‌ی آزمون شامل نمونه‌های هر پنج نوع خطا محاسبه شد و سپس نتایج برای هر کلاس به‌صورت جداگانه مورد تحلیل قرار گرفت. در ادامه، تأثیر هر یک از خطاها بر دقت مدل و نقاط قوت و ضعف آن تشریح خواهد شد.



شکل 3-1 ماتریس آشفتگی (a)



شکل 3-2 ماتریس آشفتگی(b)

Support	F1-Score	Recall	Precision	Class
303	0.59	0.65	0.54	Normal
293	0.39	0.49	0.33	AddWeight
294	0.29	0.34	0.26	PropellerDamage_slight
294	0.12	0.06	1.00	PropellerDamage_bad
294	0.54	0.52	0.57	PressureGain_constant
1478	**0.41**			**Accuracy**
1478	0.39	0.41	0.54	**Macro avg**
1478	0.39	0.41	0.54	**Weighted avg**

جدول 3-1 نتایج شبیه سازی (classification report)

نتایج ارائه‌شده نشان‌دهنده عملکرد مدل در دسته‌بندی انواع مختلف عیوب در سیستم AUV است. در اینجا، به‌طور دقیق‌تر نتایج موجود را تجزیه و تحلیل می‌کنیم:

1. دقت(Precision)

دقت یا Precision به معنای این است که از تمامی پیش‌بینی‌هایی که مدل به عنوان یک کلاس خاص انجام داده، چه درصدی صحیح بوده است. به عبارت دیگر، دقت نشان‌دهنده صحت پیش‌بینی‌های مثبت مدل است.

- در کلاس "Normal" دقت 0.54 است، به این معنا که 54 درصد از پیش‌بینی‌هایی که مدل به عنوان "Normal" انجام داده، در واقع درست بوده است.
- در کلاس"PropellerDamage_bad" ، دقت 1.00 است که به این معنی است که هر زمان مدل پیش‌بینی کرده که آسیب شدید به پروپیلر وجود دارد، پیش‌بینی‌اش صحیح بوده است. البته، این کلاس دارای مقادیر کم در Recall است که در ادامه توضیح خواهیم داد.

2. یادآوری(Recall)

یادآوری یا Recall نشان‌دهنده این است که مدل از تمامی نمونه‌های واقعی یک کلاس خاص، چه درصدی را شناسایی کرده است. به عبارت دیگر، Recallاندازه‌گیری می‌کند که مدل چقدر قادر بوده تا تمام نمونه‌های مثبت واقعی را شناسایی کند.

- در کلاس "Normal" یادآوری 0.65 است، به این معنی که مدل توانسته 65 درصد از نمونه‌های واقعی کلاس "Normal" را شناسایی کند.
- در کلاس"PropellerDamage_bad" ، یادآوری 0.06 است، که به‌طور واضح نشان می‌دهد که مدل تنها 6 درصد از نمونه‌های آسیب شدید به پروپیلر را شناسایی کرده است.

3. امتیاز(F1-Score) F1

امتیاز F1 به‌عنوان میانگین هماهنگ دقت و یادآوری شناخته می‌شود و برای ارزیابی مدل‌های دسته‌بندی در شرایطی که کلاس‌ها نابرابر هستند بسیار مفید است. این معیار به‌ویژه زمانی که تعداد نمونه‌های یک کلاس بسیار کمتر از دیگری باشد، کاربرد دارد.

- برای کلاس"Normal" ، F1-Scoreبرابر با 0.59 است که نشان‌دهنده عملکرد نسبتاً خوب مدل در این کلاس است.
- برای کلاس"PropellerDamage_bad" ، F1-Scoreبسیار پایین است (0.12)، که به این معنی است که حتی با دقت 1.00، مدل نتواسته به‌طور مؤثر تمام نمونه‌های این کلاس را شناسایی کند.

4. دقت کلی(Accuracy)

دقت کلی مدل (Accuracy) برابر با 0.41 است، که نشان می‌دهد مدل تنها قادر به درست پیش‌بینی 41 درصد از تمامی نمونه‌ها بوده است. این میزان دقت نسبتاً پایین است که می‌تواند به دلیل وجود داده‌های نامتوازن (مقدار کمتر داده برای برخی کلاس‌ها) یا مشکلات مدل‌سازی باشد.

5. میانگین‌های ماکرو و وزنی(Macro avg & Weighted avg)

- **Macro avg:** این میانگین به‌طور ساده میانگین دقت، یادآوری و F1-Score در تمام کلاس‌ها را محاسبه می‌کند. در اینجا، Macro avg برای دقت 0.54، برای یادآوری 0.41 و برای F1-Score 0.39 است. این نتایج نشان می‌دهند که عملکرد مدل به طور کلی در همه کلاس‌ها متوسط است.
- **Weighted avg:** این میانگین وزن‌دار است که بر اساس تعداد نمونه‌های هر کلاس وزن‌دهی می‌شود. این نتایج مشابه Macro avg است، زیرا توزیع داده‌ها تقریباً یکنواخت است و تفاوت‌های زیادی میان کلاس‌ها مشاهده نمی‌شود.

تحلیل ماتریس‌های کانفیوژن:

ماتریس‌های کانفیوژن به مدل کمک می‌کنند تا دقیق‌تر بفهمیم که کجاها خطا کرده است. در اینجا، به تحلیل کلاس‌های مختلف از طریق ماتریس‌های کانفیوژن می‌پردازیم:

- **کلاس "Normal":** مدل توانسته است با دقت 0.65 به خوبی نمونه‌های این کلاس را شناسایی کند. اما نکته‌ای که باید ذکر کرد این است که در برخی از موارد، پیش‌بینی‌های اشتباهی انجام شده است.
- **کلاس "AddWeight":** مدل با دقت 0.33 تنها توانسته است به‌طور جزئی به شناسایی این کلاس بپردازد. علاوه بر این، F1-Score پایین 0.39 نشان می‌دهد که مدل در شناسایی این نوع خطا نیز مشکلاتی داشته است.
- **کلاس "PropellerDamage_slight":** مدل با دقت 0.26 این کلاس را شناسایی کرده است. با توجه به این که پیش‌بینی این کلاس برای مدل بسیار سخت بوده، دقت و یادآوری پایین آن را می‌توان به دلیل پیچیدگی شبیه‌سازی یا داده‌های محدود این کلاس دانست.
- **کلاس "PropellerDamage_bad":** این کلاس نتایج جالبی به همراه دارد. با وجود دقت 1.00، یادآوری تنها 0.06 است که به این معنی است که مدل بسیار به‌ندرت قادر به شناسایی

نمونه‌های آسیب شدید به پروپیلر بوده است. این مشکل نشان‌دهنده نادر بودن این نوع خطا در داده‌ها و یا مشکل در داده‌سازی و پیش‌پردازش است.

- **کلاس :"PressureGain_constant"** دقت 0.57 و F1-Score 0.54 برای این کلاس نشان‌دهنده عملکرد نسبی مناسب مدل در شناسایی آن است.

پیشنهادات برای بهبود مدل

1. **افزایش داده‌ها و داده‌های متوازن :**یکی از دلایل اصلی عملکرد ضعیف مدل در شناسایی برخی کلاس‌ها، به‌ویژه"PropellerDamage_bad" ، می‌تواند عدم توازن داده‌ها باشد. افزایش داده‌های مربوط به کلاس‌های نادر و استفاده از تکنیک‌هایی مانند oversampling یا undersamplingمی‌تواند به بهبود عملکرد مدل کمک کند.
2. **بهبود پیش‌پردازش داده‌ها :**پیش‌پردازش مناسب داده‌ها می‌تواند به مدل کمک کند تا ویژگی‌های مهم را بهتر شناسایی کند. برای مثال، نرمال‌سازی و یا انتخاب ویژگی‌های بهتر می‌تواند تاثیر قابل توجهی داشته باشد.
3. **استفاده از مدل‌های پیچیده‌تر :**استفاده از شبکه‌های عصبی عمیق‌تر و مدل‌های پیچیده‌تر می‌تواند توانایی مدل را در شناسایی الگوهای پیچیده‌تر افزایش دهد.
4. **تنظیمات مدل :**بهینه‌سازی و تنظیم هایپرامترهای مدل) مانندlearning rate ، تعداد لایه‌ها و نورون‌ها، و (... می‌تواند عملکرد مدل را به طور چشمگیری بهبود بخشد.
5. **استفاده از تکنیک‌های کاهش خطا :**برای بهبود دقت مدل در کلاس‌های خاص، می‌توان از تکنیک‌های اختصاصی کاهش خطا مانند کلاس‌های تعادلی (class balancing) استفاده کرد.

فصل چهارم:

عیب یابی مبتنی بر روش DBN

معرفی روش DBN

شبکه‌های بیزی عمیق (DBN) یکی از مدل‌های یادگیری عمیق است که برای آموزش ویژگی‌ها و نمایش‌های غیرخطی پیچیده از داده‌ها طراحی شده است DBN .یک مدل سلسله‌مراتبی از لایه‌های مخفی است که هر لایه از آن‌ها به صورت یک شبکه بیزی در نظر گرفته می‌شود. این مدل برای یادگیری ویژگی‌های پیچیده از داده‌ها به کار می‌رود و به دلیل توانایی در مدل‌سازی داده‌ها با ابعاد بالا، در مسائل مختلف مانند پردازش تصویر، پردازش زبان طبیعی و یادگیری ماشین بسیار مورد توجه قرار گرفته است.

ساختار DBN

DBNاز چندین لایه خودسازگار تشکیل شده است که شامل یک لایه ورودی و چندین لایه مخفی می‌باشد. هر لایه مخفی از یک مدل بیزی) مانند (Restricted Boltzmann Machines - RBM استفاده می‌کند که به آن این قابلیت را می‌دهد که روابط پیچیده میان ویژگی‌ها را در داده‌ها بیاموزد. لایه‌های مخفی در DBN به‌طور سلسله‌مراتبی از ویژگی‌های ساده‌تر به ویژگی‌های پیچیده‌تر پردازش می‌کنند و در نهایت یک نمایش فشرده از داده‌ها را تولید می‌کنند.

فرآیند آموزش

آموزش DBN معمولاً به صورت دو مرحله‌ای انجام می‌شود:

1. **آموزش پیش‌گذر :(Pretraining)** در این مرحله، مدل ابتدا به صورت غیرنظارتی (unsupervised)آموزش داده می‌شود. این آموزش معمولاً با استفاده ازRBM ها انجام می‌شود که به مدل کمک می‌کند تا ویژگی‌های ابتدایی داده‌ها را یاد بگیرد. این مرحله به مدل اجازه می‌دهد تا ویژگی‌های پنهانی را شبیه‌سازی کند بدون اینکه به برچسب‌های خروجی نیاز داشته باشد.
2. **آموزش نظارتی :(Supervised Fine-Tuning)** پس از پیش‌آموزش، شبکه به صورت نظارتی با استفاده از داده‌های برچسب‌خورده آموزش داده می‌شود. این مرحله به مدل کمک می‌کند تا ارتباطات بین ویژگی‌های یادگرفته‌شده و کلاس‌های هدف را مدل‌سازی کند.

مزایایDBN

- ✓ **یادگیری ویژگی‌های پیچیده** DBN **:**قادر است به‌طور مؤثر ویژگی‌های پیچیده و غیرخطی را از داده‌های ورودی استخراج کند.
- ✓ **یادگیری غیرنظارتی :**مدل می‌تواند به صورت غیرنظارتی (بدون نیاز به برچسب‌های خروجی) ویژگی‌های داده‌ها را یاد بگیرد.
- ✓ **قابلیت مقیاس‌پذیری**DBN **:**ها می‌توانند به‌راحتی با داده‌های با ابعاد بالا و پیچیده مقیاس پیدا کنند.

کاربردها

DBNها در بسیاری از زمینه‌های یادگیری ماشین و یادگیری عمیق کاربرد دارند، از جمله:

- ✓ پردازش تصویر و شناسایی الگو
- ✓ پردازش زبان طبیعی
- ✓ یادگیری ویژگی‌های پیچیده در داده‌های بزرگ
- ✓ سیستم‌های توصیه‌گر و تحلیل داده‌های حجیم

به‌طور کلی، DBNها به‌عنوان یک مدل قوی در یادگیری عمیق شناخته می‌شوند که توانایی پردازش داده‌ها و استخراج ویژگی‌های پنهان را به‌طور مؤثر دارا هستند.

پردازش داده‌ها و آماده‌سازی آن‌ها

ابتدا داده‌ها از چندین فایل CSV بارگذاری شدند که هر فایل مربوط به یک وضعیت مختلف سیستم است. در این پروژه، پنج وضعیت مختلف داریم:

- **Normal**(وضعیت نرمال)
- **AddWeight**(افزایش وزن)
- **PropellerDamage_slight**(آسیب جزئی به پروانه)
- **PropellerDamage_bad**(آسیب شدید به پروانه)
- **PressureGain_constant**(افزایش فشار ثابت)

برای هر کدام از این وضعیت‌ها، 50 فایل داده از نوع CSV با تعداد مشخصی از ویژگی‌ها (شامل اندازه‌گیری‌های مختلف از سیستم) داشتیم. این داده‌ها به یک آرایه numpy تبدیل شدند تا برای آموزش و ارزیابی مدل قابل استفاده باشند.

ایجاد برچسب‌ها(Labels)

برای انجام شبیه‌سازی‌های تشخیص، نیاز به برچسب‌گذاری داده‌ها داریم. به این ترتیب که هر نمونه داده به یک کلاس خاص اختصاص می‌یابد:

- کلاس 0 برای **Normal**
- کلاس 1 برای **AddWeight**
- کلاس 2 برای **PropellerDamage_slight**
- کلاس 3 برای **PropellerDamage_bad**
- کلاس 4 برای **PressureGain_constant**

این برچسب‌ها برای هر کدام از داده‌ها مشخص شده و در نهایت، داده‌ها و برچسب‌ها به آرایه‌های ورودی و خروجی مدل تقسیم شدند.

پیش‌پردازش داده‌ها

در مرحله پیش‌پردازش، از **MinMaxScaler**برای مقیاس‌بندی داده‌ها استفاده شد. این تکنیک به این منظور استفاده می‌شود که تمامی ویژگی‌های داده‌ها در دامنه‌ای ثابت و مشخص قرار بگیرند (بین 0 و 1). این عمل برای بهبود عملکرد مدل‌های یادگیری عمیق، که حساس به مقیاس داده‌ها هستند، انجام شد.

ساخت و آموزش مدل‌های(RBM (Restricted Boltzmann Machine

در اینجا، از **RBM (Restricted Boltzmann Machine)**به عنوان یک مدل پیش‌آموزش برای ویژگی‌های داده‌ها استفاده کردیمRBM .ها مدل‌های غیرقابل نظارت هستند که می‌توانند ویژگی‌های جدید و مفیدی از داده‌ها استخراج کنند، به‌ویژه زمانی که داده‌ها دارای ابعاد زیادی باشند.

دلایل استفاده ازRBM :

- **پیش‌آموزش ویژگی‌ها**RBM **:**ها به صورت خودکار ویژگی‌های جدیدی از داده‌ها استخراج می‌کنند که می‌تواند برای تشخیص بهتر در مدل‌های بعدی استفاده شود.
- **کاهش ابعاد :**با استفاده ازRBM ها می‌توان ابعاد داده‌ها را کاهش داد، که این کار به کاهش پیچیدگی مدل کمک می‌کند.

در این بخش، ابتدا برای هر لایه از مدل، پیش‌آموزش صورت می‌گیرد که در آن داده‌ها به کمک **contrastive divergence**به روز می‌شوند.

ساخت مدل DBN (Deep Belief Network)

بعد از پیش‌آموزش ویژگی‌ها باRBM ، داده‌های استخراج‌شده به لایه‌های بعدی شبکه ارسال می‌شوند . **DBN**ترکیبی از چندین RBM اســـت که می‌تولند ویژگی‌های پیچیده‌تری از داده‌ها یاد بگیرد و در نتیجه عملکرد بهتری در تشخیص مشکلات سیستم‌ها داشته باشد.

دلیل انتخاب DBN :

- **یادگیری غیر نظارتی** DBN **:**از شـــبکه‌های RBM برای یادگیری ویژگی‌های پیچیده داده‌ها استفاده می‌کند، که می‌تواند به مدل کمک کند تا به طور دقیق‌تری به مشکلات پی‌ببرد.
- **ساختار چند لایه :**این شبکه به صورت سلسله‌مراتبی از ویژگی‌ها یاد می‌گیرد، بنابراین قادر به شبیه‌سازی رفتار پیچیده‌تری از سیستم است.

آموزش نهایی مدل با استفاده از Cross-Entropy Loss

در مرحله نهایی، پس از پیش‌آموزش با اســـتفاده ازRBM ، از یک **طبقه‌بند خطی** برای تقســـیم‌بندی داده‌ها به کلاس‌های مختلف استفاده می‌شود. سپس، مدل با استفاده از تابع هزینه **Cross-Entropy Loss**آموزش داده می‌شود. این تابع برای مسائل طبقه‌بندی چندکلاسه مناسب است و به‌طور مستقیم به کاهش اشتباهات طبقه‌بندی کمک می‌کند.

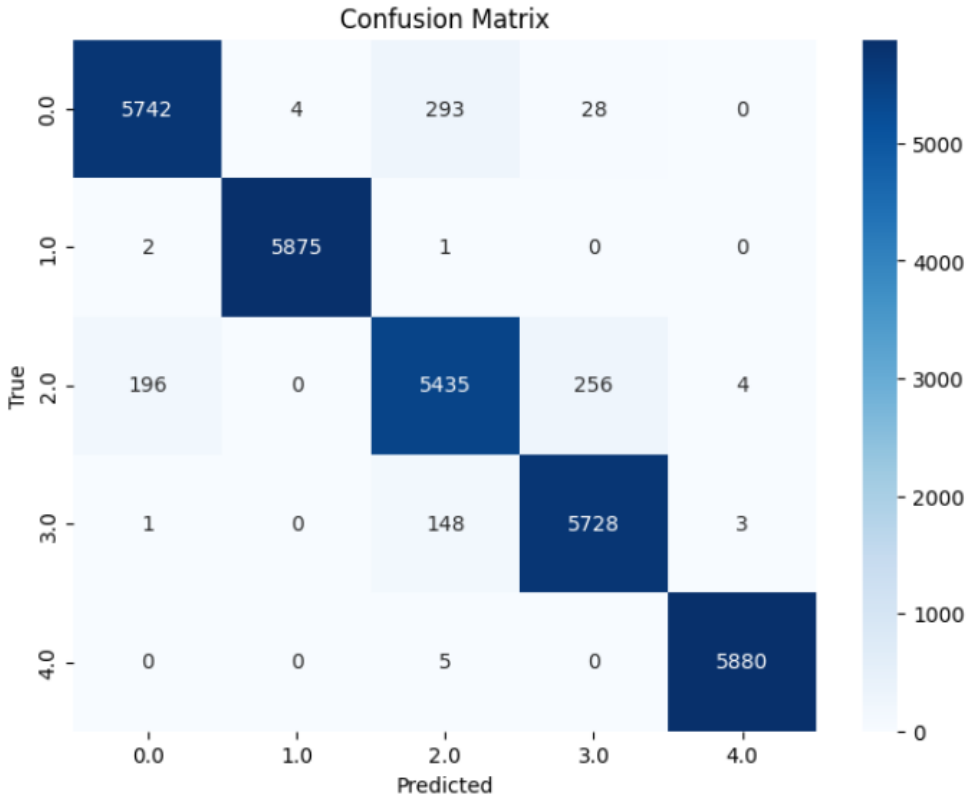
عددگذاری‌ها(Hyperparameter Tuning)

برای آموزش مدل، از چندین عددگذاری مختلف اســتفاده شــد که شــامل انتخاب اندازه دســته (batch size)، نرخ یادگیری(learning rate) ، و تعداد اپوک‌ها (epochs) می‌باشد:

- **نرخ یادگیری Learning Rate) :**(نرخ یادگیری 0.1 برای پیش‌آموزشRBM ها و 0.001 برای آموزش نهایی مدل استفاده شد. این انتخاب‌ها بر اساس آزمایش‌های مختلف بهینه شدند.
- **Batch Size:**در این مدل، سایز دسته‌ها 64 در نظر گرفته شد تا مدل بتواند به‌طور مؤثری یاد بگیرد و زمان آموزش کوتاه‌تری داشته باشد.
- **Epochs:**تعداد اپوک‌ها 1000 در نظر گرفته شده است.

نتایج شبیه سازی

در این بخش به تحلیل کمی و کیفی عملکرد مدل در تشـخیص حالات مختلف سـیسـتم می‌پردازیم. معیارهای اصلی سنجش ما «ماتریس آشفتگی (Confusion Matrix) «و «دقت کلی (Accuracy) « هستند. ماتریس آشفتگی امکان بررسی جزئی نواحی صحیح و ناصحیح طبقه‌بندی را برای هر یک از کلاس‌های پنج‌گانه فراهم می‌کند و نشان می‌دهد مدل در شناسایی نمونه‌های هر وضعیت تا چه اندازه موفق یا دچار خطا بوده است. دقت کلی نیز درصد پیش‌بینی‌های صحیح را در کل نمونه‌های آزمون ارائه می‌کند و معیاری یک‌بعدی برای سنجش توانایی عمومی مدل محسوب می‌شود. در ادامه، ابتدا ماتریس آشفتگی را برای هر کلاس بررسی کرده و الگوهای خطا را شناسایی می‌کنیم، سپس دقت کلی را گزارش و با اهداف مدنظر مقایسه خواهیم کرد. نهایتاً نقاط قوت و ضعف مدل بر اساس این دو معیار استخراج شده و راهکارهایی برای بهبود عملکرد ارائه خواهد شد.



شکل 4-1 ماتریس آشفتگی

- ✓ Accuracy: 0.9682

نتایج به‌دست‌آمده از ماتریس آشفتگی نشان‌دهندهٔ عملکرد برجستهٔ مدل در شبیه‌سازی و دسته‌بندی حالات مختلف است. هر کلاس توانسته است درصد بالایی از دقت را در شبیه‌سازی خود به‌دست آورد که به تفصیل به شرح زیر است:

1. **کلاس 0 :(Normal)** دقت **95.7%** در این کلاس به معنای آن است که اکثر نمونه‌ها به درستی به عنوان "حالت نرمال" شناسایی شده‌اند. این عملکرد نشان‌دهندهٔ تمایز قوی بین ویژگی‌های حالت نرمال و دیگر حالات است.
2. **کلاس 1 :(AddWeight)** دقت **تقریباً 100 %** در این کلاس نشان‌دهندهٔ تمایز بسیار عالی بین حالت افزودن وزن و سایر حالت‌ها است. مدل به‌طور مؤثر ویژگی‌های منحصر به فرد این حالت را شناسایی کرده و تقریباً تمام نمونه‌ها را به درستی دسته‌بندی کرده است.
3. **کلاس 2 :(PropellerDamage_slight)** دقت **95.3%** در این کلاس به این معناست که اکثر نمونه‌ها به درستی در این دسته قرار گرفته‌اند، که نشان‌دهندهٔ این است که ویژگی‌های آسیب جزئی پروانه به‌خوبی تفکیک شده‌اند، اگرچه مقداری اشتباه‌پذیری در دسته‌بندی وجود دارد.
4. **کلاس 3 :(PropellerDamage_bad)** دقت **96%** در این کلاس به معنای عملکرد بسیار خوب مدل در شناسایی آسیب شدید پروانه است. تقریباً تمامی نمونه‌های این کلاس به درستی شناسایی شده‌اند و این موضوع بیانگر قدرت مدل در تفکیک دقیق حالات مشابه به هم است.
5. **کلاس 4 :(PressureGain_constant)** دقت **تقریباً 100 %** در این کلاس به معنای آن است که مدل به‌طور کامل و با دقت بالا قادر به شناسایی خرابی سنسور عمق بوده است. تقریباً تمام نمونه‌های این کلاس به درستی دسته‌بندی شده‌اند.

تحلیل کلی:

- **دقت بالای مدل در هر کلاس** (بیش از 95٪) نشان‌دهندهٔ قابلیت مدل در شناسایی دقیق ویژگی‌های رفتاری هر حالت است.

- **تنوع و پیچیدگی حالات مختلف** (مثل آسیب پروانه با شدت‌های مختلف) به‌خوبی از هم تفکیک شده‌اند و این نشان می‌دهد که شبکه در یادگیری ویژگی‌های پیچیده به‌خوبی عمل کرده است.
- **پرفورمنس کلی مدل** در تشخیص و شبیه‌سازی حالات مختلف بسیار قوی است و این نکته به‌ویژه در دقت‌های نزدیک به 100٪ در کلاس‌هایی مانند "AddWeight" و "PressureGain_constant"مشهود است.

در نتیجه، مدل با دقت بالا در شناسایی و تفکیک حالات مختلف، نتایج مطلوبی ارائه داده است که نشان از عملکرد مطلوب در شبیه‌سازی سیستم دارد.

فصل پنجم:

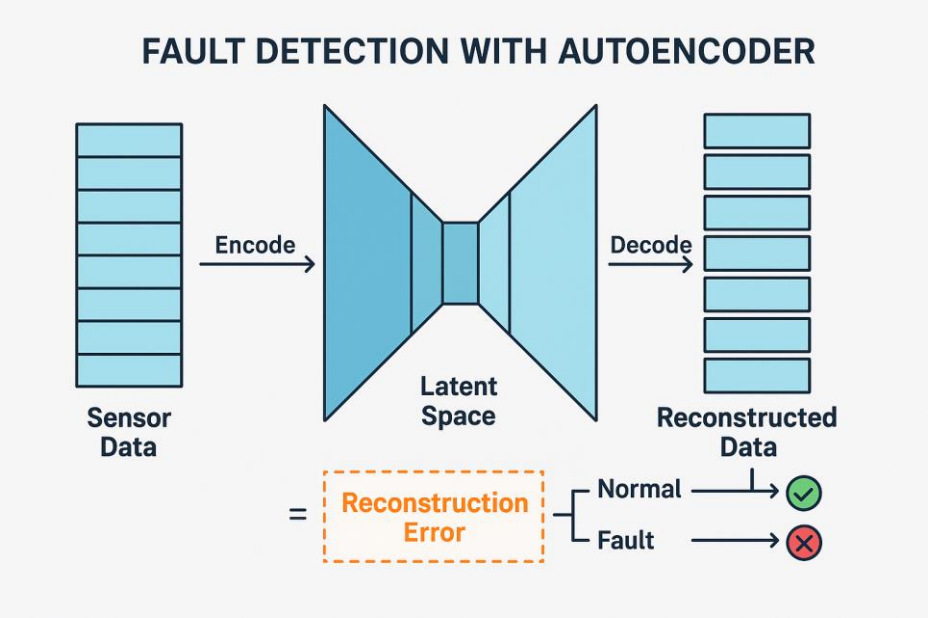
عیب‌یابی مبتنی بر بازسازی داده‌ها با استفاده ازAutoencoder

معرفی روش

در بسیاری از سامانه‌های صنعتی و سیستم‌های پیچیده، پایش وضعیت و تشخیص زودهنگام خطاها نقش مهمی در کاهش هزینه‌های نگهداری، افزایش ایمنی، و بهبود عملکرد ایفا می‌کند. یکی از روش‌های نوین و مؤثر در این زمینه، استفاده از **شبکه‌های عصبی Autoencoder** برای **تشخیص ناهنجاری** در داده‌های حاصل از حسگرهاست.

Autoencoderنوعی شبکه عصبی بدون ناظر است که با هدف بازسازی ورودی خود طراحی می‌شود. این شبکه ابتدا داده‌ها را به فضای نهفته (Latent Space) فشرده‌سازی کرده و سپس تلاش می‌کند آن‌ها را بازسازی کند. اگر مدل فقط بر روی داده‌های نرمال آموزش ببیند، تولیایی بازسازی داده‌های ناهنجار یا غیرعادی را به خوبی نخواهد داشت. بر همین اساس، **اختلاف بین داده‌ی ورودی و خروجی بازسازی‌شده (خطای بازسازی)** به‌عنوان یک شاخص برای تشخیص ناهنجاری به کار گرفته می‌شود.

در این پروژه، با آموزش Autoencoder تنها بر روی داده‌های نرمال، یک آستانه (Threshold) برای خطای بازسازی تعیین می‌شود. سپس داده‌های تست با این مدل پردازش شده و آن‌هایی که خطای بازسازی بالاتری نسبت به آستانه دارند، به‌عنوان داده‌های مشکوک یا معیوب شناسایی می‌گردند. این روش نه‌تنها نیاز به برچسب‌گذاری گسترده ندارد، بلکه برای محیط‌هایی که داده‌های خطا بسیار کمیاب یا ناشناخته هستند، بسیار مناسب است.



شکل 5-1 ساختار روش عیب یابی با استفاده از Auto Encoder

پیش‌پردازش و آماده‌سازی داده‌ها برای آموزش مدل تشخیص خطا

در این بخش، داده‌های خام حاصل از پنج وضعیت عملکرد مختلف سیستم شامل عملکرد نرمال، بار اضافی، آسیب خفیف پروانه، آسیب شدید پروانه و افزایش ثابت فشار جمع‌آوری شده‌اند. هدف اصلی از آماده‌سازی این داده‌ها، ایجاد بستری مناسب برای آموزش و ارزیابی یک مدل تشخیص ناهنجاری مبتنی بر یادگیری بدون نظارت است.

داده‌های مربوط به هر وضعیت، به‌صورت مجزا برچسب‌گذاری شده‌اند تا در مراحل ارزیابی عملکرد مدل، امکان تفکیک ناهنجاری‌ها از داده‌های سالم فراهم باشد. در راستای تمرکز بر داده‌های نرمال برای آموزش مدلAutoencoder ، داده‌های نرمال به‌صورت جداگانه استخراج و به‌عنوان تنها ورودی مدل در نظر گرفته شده‌اند. این رویکرد، مبتنی بر اصل بازسازی بهینه داده‌های سالم توسط Autoencoder است، در حالی که داده‌های ناهنجار به دلیل تفاوت الگوی رفتار، خطای بازسازی بیشتری خواهند داشت.

در گام بعد، برای حذف تأثیر تفاوت مقیاس بین ویژگی‌ها، کلیه داده‌ها با استفاده از روش استانداردسازی نرمال شده‌اند. در این روش، ویژگی‌ها با کم‌کردن میانگین و تقسیم بر انحراف معیار، به بازه‌ای با میانگین

صفر و واریانس یک منتقل می‌شوند. نرمال‌سازی، نقش مهمی در پایداری فرایند آموزش و بهبود کیفیت تشخیص ایفا می‌کند.

در نهایت، داده‌های آزمون شامل ترکیبی از کلیه وضعیت‌های عملکرد سیستم هستند که برای ارزیابی قابلیت تشخیص ناهنجاری مدل مورد استفاده قرار می‌گیرند.

مدل Autoencoder مبتنی بر شبکه‌های عصبی چندلایه (MLP) برای تشخیص ناهنجاری

در این بخش، یک مدل Autoencoder مبتنی بر شبکه عصبی پرسپترون چندلایه (MLP) به‌منظور تشخیص ناهنجی در داده‌های عملکرد سیستم طراحی و پیاده‌سازی شده است. هدف اصلی از به‌کارگیری این معماری، یادگیری الگوهای رفتاری داده‌های نرمال و استفاده از خطای بازسازی به‌عنوان شاخصی برای شناسایی داده‌های غیرعادی است.

معماری مدل شامل چندین لایه‌ی متوالی در بخش‌های رمزگذار (encoder) و رمزگشا (decoder) است. در رمزگذار، داده‌ی ورودی به فضایی با ابعاد کمتر فشرده می‌شود و در رمزگشا تلاش می‌شود که این بردار فشرده‌شده به شکل اولیه بازسازی گردد. برای جلوگیری از بیش‌برازش و بهبود توان تعمیم مدل، از لایه‌های ریزش (dropout) نیز استفاده شده است.

آموزش مدل تنها با استفاده از داده‌های نرمال انجام شده است، زیرا فرض بر این است که داده‌های ناهنجار (حاصل از شرایط غیرعادی عملکرد سیستم) الگوی متفاوتی نسبت به داده‌های نرمال دارند و در نتیجه، بازسازی ضعیف‌تری از آن‌ها حاصل خواهد شد. به همین دلیل، تفاوت بین داده‌ی ورودی و خروجی بازسازی‌شده (یعنی خطای بازسازی) می‌تواند به‌عنوان معیار تشخیص ناهنجاری به‌کار رود.

برای تعیین آستانه تشخیص ناهنجاری، از روش آماری مبتنی بر میانگین و انحراف معیار خطای بازسازی داده‌های نرمال استفاده شده است. سپس با استفاده از این آستانه، نمونه‌های تست به دو گروه نرمال و ناهنجار تفکیک شده‌اند. در نهایت، با بهره‌گیری از معیارهایی چون دقت و یادآوری، عملکرد مدل در تشخیص ناهنجاری‌ها ارزیابی شده است.

نتایج مدل اول

در این بخش، نتایج ارزیابی مدل تشخیص ناهنجاری مبتنی بر Autoencoder که با استفاده از داده‌های پیش‌پردازش‌شده آموزش دیده است، آورده شده است. مدل به منظور شبیه‌سازی رفتار ناهنجاری‌ها از داده‌های نرمال آموزش دیده و سپس عملکرد آن در شناسایی ناهنجاری‌ها مورد ارزیابی قرار گرفت.

نتایج حاصل از ارزیابی مدل به شرح زیر است:

ماتریس سردرگمی:

ماتریس سردرگمی حاصل نشان‌دهنده تعداد پیش‌بینی‌های صحیح و اشتباه برای هر کلاس است. در این ماتریس، ردیف اول نشان‌دهنده پیش‌بینی‌ها برای داده‌های نرمال و ردیف دوم مربوط به پیش‌بینی‌ها برای داده‌های ناهنجار است. مقدار بالای صحیح بودن پیش‌بینی‌ها در دسته‌بندی داده‌های ناهنجار (15637 پیش‌بینی صحیح در برابر 3972 اشتباه) و همچنین دقیق بودن پیش‌بینی‌های مربوط به داده‌های نرمال (4974 پیش‌بینی صحیح در برابر 110 اشتباه) بیانگر دقت بالای مدل در تشخیص ناهنجاری‌ها و رفتار نرمال است.

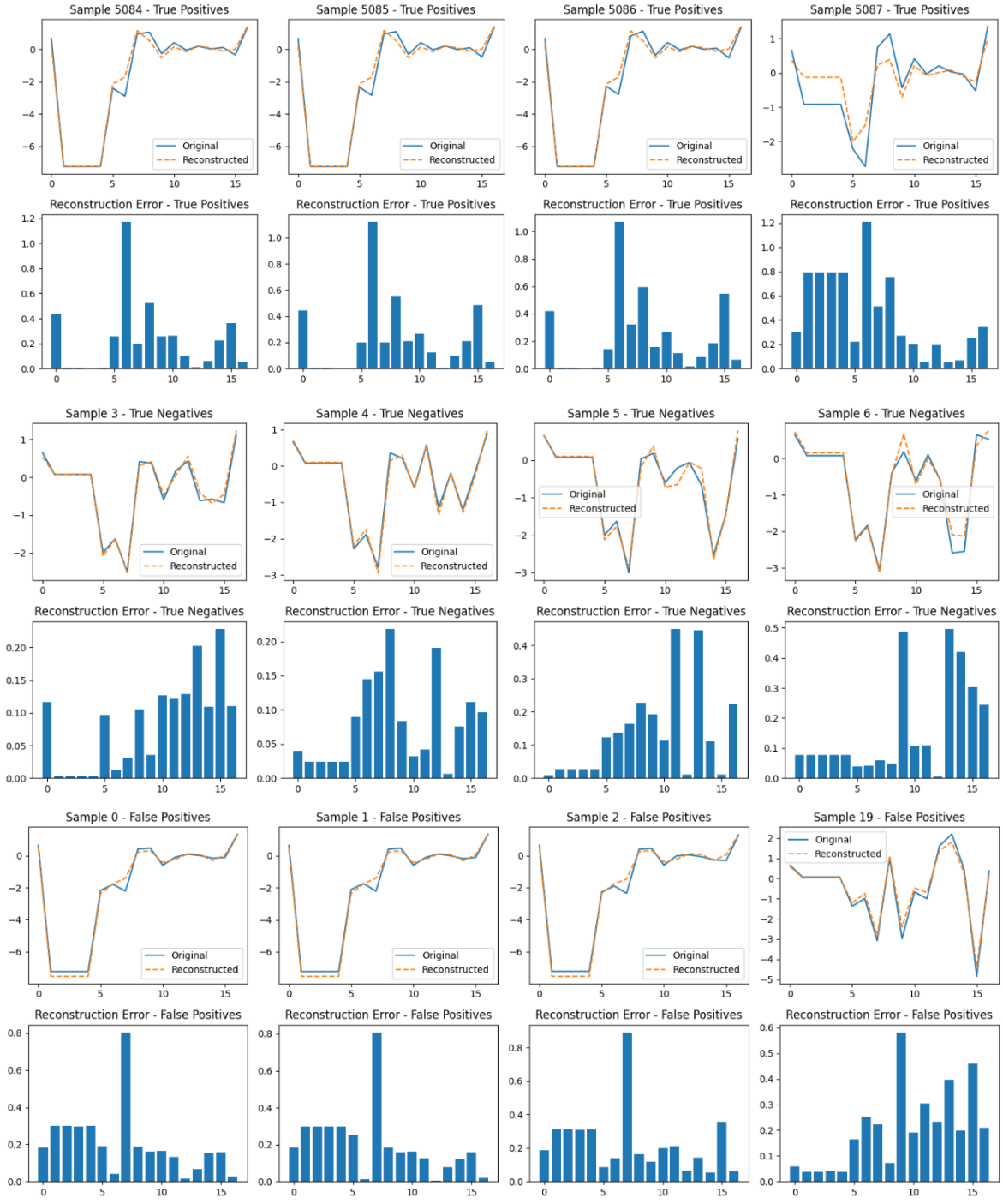
دقت:(Precision)

دقت مدل برابر با 0.99 است که نشان‌دهنده آن است که از تمام پیش‌بینی‌های ناهنجار مدل، 99 درصد آنها صحیح بوده‌اند. این مقدار بالا بیانگر آن است که مدل توانسته است به‌طور مؤثر ویژگی‌های ناهنجاری را شناسایی کرده و تعداد زیادی از نمونه‌های ناهنجار را به‌طور دقیق تشخیص دهد.

بازخوانی:(Recall)

بازخوانی مدل 0.80 است که نشان‌دهنده آن است که 80 درصد از تمامی نمونه‌های ناهنجار به درستی شناسایی شده‌اند. این مقدار نشان‌دهنده قدرت مدل در شناسایی ناهنجاری‌ها و توانایی آن در تشخیص بیشتر موارد ناهنجار است.

نتایج به‌دست‌آمده از مدل، کارایی بالای آن را در شبیه‌سازی و تشخیص رفتار ناهنجار در سیستم مورد نظر تایید می‌کند. این ارزیابی نشان می‌دهد که مدل توانسته است ویژگی‌های مؤثر از سیگنال‌های فرکانسی استخراج‌شده را شناسایی کند و ناهنجاری‌ها را با دقت قابل قبولی تفکیک نماید.



شکل 5-2 نتایج شبیه سازی مدل اول

در این بخش، برای تحلیل عملکرد مدل تشخیص ناهنجاری، گراف‌هایی از نمونه‌های مختلف پیش‌بینی‌شده از انواع نتایج ماتریس سردرگمی(True Positives) ، True Negatives، False Positives، و False Negatives)ترسیم شده است. این گراف‌ها شامل دو بخش اصلی هستند: داده‌های اصلی و بازسازی‌شده (reconstructed data)و همچنین خطای بازسازی برای هر نمونه.

داده‌های اصلی و بازسازی‌شده :در این گراف‌ها، داده‌های اصلی که به مدل وارد شده‌اند (با خط پیوسته) و داده‌های بازسازی‌شده توسط Autoencoder (با خط نقطه‌چین) نمایش داده شده‌اند. برای هر نوع نتیجه پیش‌بینی، تا چهار نمونه به‌طور تصادفی انتخاب شده‌اند تا نشان دهند مدل چگونه سیگنال‌های نرمال و ناهنجار را بازسازی کرده است. این گراف‌ها نشان می‌دهند که مدل توانسته است داده‌های نرمال را به‌طور دقیق بازسازی کند و تفاوت‌های جزئی را در داده‌های ناهنجار شبیه‌سازی نماید.

خطای بازسازی :در قسمت دوم هر گراف، اختلاف مطلق بین داده‌های اصلی و بازسازی‌شده (که به‌عنوان خطای بازسازی شناخته می‌شود) به‌صورت میله‌ای نمایش داده شده است. این خطا نشان می‌دهد که مدل تا چه اندازه قادر به بازسازی دقیق داده‌ها بوده است. برای داده‌های نرمال، خطای بازسازی معمولاً کم است، در حالی که برای داده‌های ناهنجار، خطای بازسازی به‌طور قابل توجهی بیشتر است.

نتایج مطلوب:

True Positives (TP):برای داده‌های ناهنجار که به‌درستی به عنوان ناهنجار شناسایی شده‌اند، خطای بازسازی بیشتر از داده‌های نرمال است که نشان‌دهنده عملکرد صحیح مدل در شناسایی ناهنجاری‌ها است.

True Negatives (TN):برای داده‌های نرمال که به‌درستی به عنوان نرمال شناسایی شده‌اند، بازسازی دقیق بوده و خطای بازسازی کمتر است.

False Positives (FP)و **False Negatives (FN)**:مدل در شناسایی ناهنجاری‌ها یا داده‌های نرمال با دقت مناسبی عمل کرده است، به‌طوری‌که خطای بازسازی در هر دو حالت به‌طور معناداری تفاوت دارد و مدل توانسته است اکثر ناهنجاری‌ها را شناسایی کند.

با توجه به گراف‌ها و نتایج به‌دست‌آمده، می‌توان نتیجه گرفت که مدل در تشخیص ناهنجاری‌ها به‌طور مطلوب عمل کرده است و ویژگی‌های داده‌های نرمال و ناهنجار را به‌خوبی از یکدیگر تفکیک کرده است. این تحلیل تأکید بر کارایی بالا و دقت مدل در بازسازی داده‌ها و شناسایی ناهنجاری‌ها دارد.

مدل Autoencoder Variational (VAE) برای شناسایی ناهنجاری‌ها و ارزیابی

در این بخش، از مدل Autoencoder Variational (VAE) برای شناسایی ناهنجاری‌ها در داده‌ها استفاده شده است VAE .به‌عنوان یک مدل یادگیری عمیق قادر است تا به‌طور خودکار ویژگی‌های داده‌ها را استخراج کند و از این ویژگی‌ها برای شبیه‌سازی داده‌ها استفاده نماید. هدف این است که داده‌های نرمال به‌درستی بازسازی شوند و داده‌های ناهنجار تفاوت بیشتری در بازسازی نشان دهند.

ساختار مدل:VAE

1. **کدگذار(Encoder)** :

- ورودی مدل، داده‌هایی با ابعاد 17 است. کدگذار این داده‌ها را از طریق دو لایه Dense به ابعاد کوچکتری می‌برد تا ویژگی‌های پنهان داده‌ها را استخراج کند.
- در نهایت، دو لایه خروجی برای تولید میانگین (z_mean) و انحراف معیار لگاریتمی (z_log_var)برای پارامترهای توزیع احتمال از طریق تکنیک بازآرایی (Reparameterization Trick)استفاده می‌شود.
- با استفاده از این پارامترها، نمونه‌هایی از فضای پنهان (latent space) مدل استخراج می‌شوند.

2. **رمزگذار(Decoder)** :

- کدگذار از نمونه‌های فضای پنهان به‌عنوان ورودی استفاده می‌کند و آن‌ها را به ابعاد اصلی داده‌ها بازسازی می‌کند.
- ساختار رمزگذار شامل دو لایه Dense است که به‌طور نهایی به تعداد ورودی‌های اصلی داده‌ها می‌انجامد.

3. **مدلVAE** :

- مدل VAE به‌طور کلی شامل کدگذار و رمزگذار است که به‌وسیله یک مدل اصلی به هم متصل شده‌اند.

- برای فرآیند آموزش، از تکنیک بازآرایی استفاده می‌شود تا فرآیند یادگیری متناسب با ویژگی‌های داده‌های نرمال و ناهنجار باشد.

فرآیند آموزش و ارزیابی مدل:

1. **آموزش مدلVAE** :

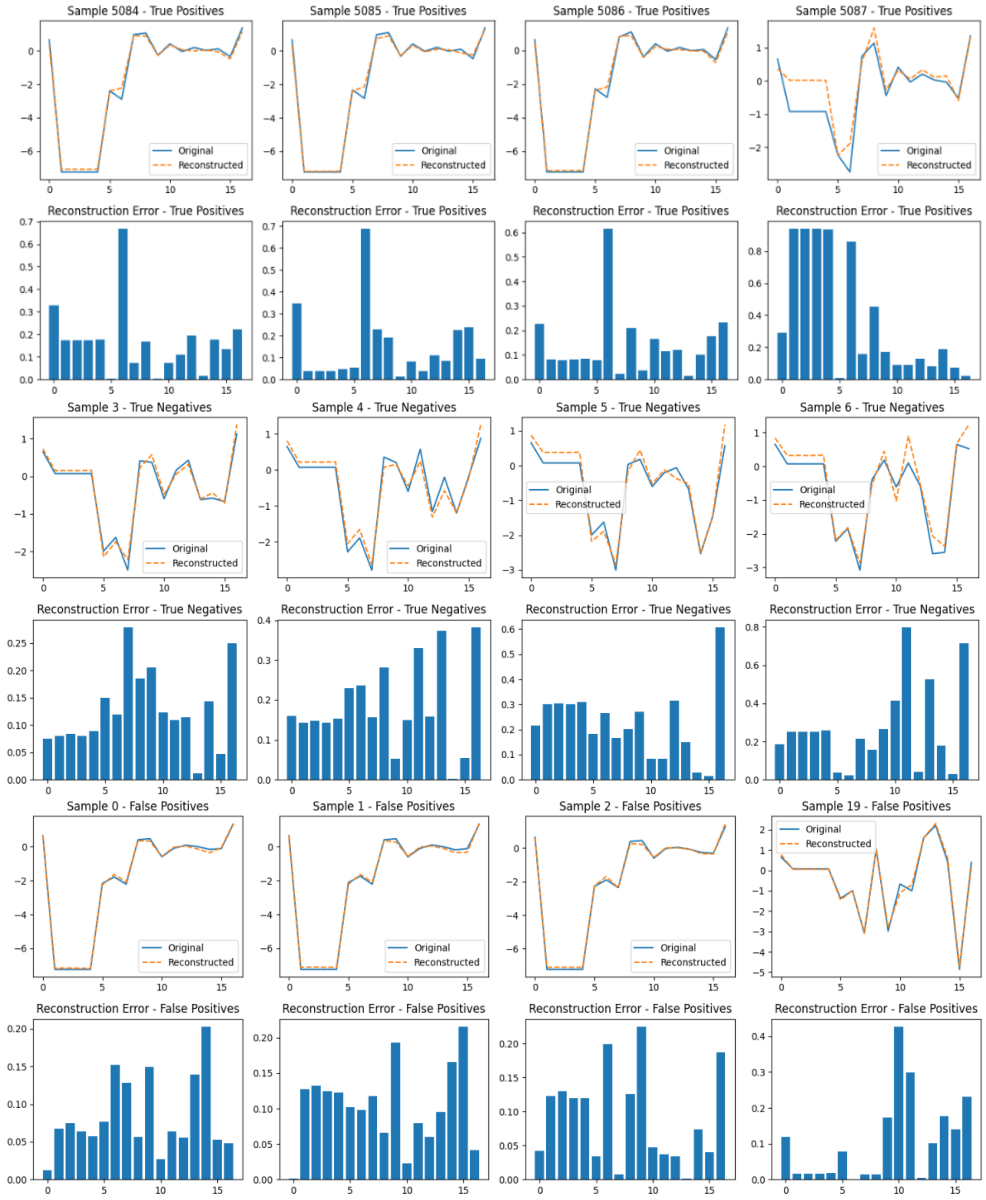
- مدل VAE با استفاده از داده‌های نرمال آموزش می‌بینند. در طی آموزش، دو نوع خطا محاسبه می‌شود:
  - **خطای بازسازی (Reconstruction Loss)** : تفاوت بین داده‌های اصلی و داده‌های بازسازی‌شده توسط رمزگذار.
  - **خطای KL Divergence**: میزان تفاوت بین توزیع واقعی داده‌ها و توزیع فرضی مدل.
- در اینجا از میانگین مربع خطا (MSE) به‌عنوان تابع هزینه برای به‌حداقل‌رساندن اختلافات بین داده‌های اصلی و بازسازی‌شده استفاده می‌شود.

نتایج مدل دوم

نتایج به‌دست‌آمده از مدل VAE نشان‌دهنده عملکرد موفق آن در تشخیص ناهنجاری‌ها است. ماتریس سردرگمی مدل حاکی از این است که مدل توانسته است بیشتر نمونه‌های ناهنجار و نرمال را به‌درستی شناسایی کند. از مجموع 15,637 نمونه ناهنجار، مدل توانسته است 3,972 نمونه ناهنجار را نادرست شناسایی کند(False Negatives) ، در حالی که فقط 110 نمونه نرمال را به‌طور نادرست ناهنجار تشخیص داده است(False Positives) .

دقت مدل 0.99 است، که نشان‌دهنده آن است که از هر 100 نمونه‌ای که مدل به‌عنوان ناهنجار شناسایی می‌کند، 99 نمونه آن‌ها به‌درستی ناهنجار هستند. این نتیجه نشان‌دهنده دقت بالای مدل در شناسایی ناهنجاری‌ها است. بازخوانی مدل نیز برابر با 0.80 است، به این معنا که مدل توانسته است 80 درصد از نمونه‌های ناهنجار را شناسایی کند. با این حال، هنوز حدود 20 درصد از ناهنجاری‌ها شناسایی نشده‌اند.

در مجموع، مدل VAE در تشخیص ناهنجاری‌ها عملکرد خوبی داشته است. این مدل به‌ویژه در شناسایی نمونه‌های ناهنجار دقیق اســـت، ولی برای بهبود بیشـــتر، می‌توان بر کاهش تعداد نمونه‌های False Negativesو افزایش بازخوانی تمرکز کرد.



شکل 5-3 نتایج شبیه سازی مدل دوم

در این بخش، تصویری که نشان‌دهنده تجزیه و تحلیل نمونه‌های مختلف پیش‌بینی مدل VAE است، ارائه می‌شود. این تصاویر به‌طور خاص به بررسی نمونه‌های مختلف طبق ماتریس سردرگمی پرداخته‌اند و به تفکیک لنواع پیش‌بینی‌ها یعنی(TP) True Positives ، (TN) True Negatives، False Positives (FP)و False Negatives (FN) نشان داده شده‌اند.

True Positives (TP):در این بخش، نمونه‌هایی که به‌درستی به عنوان ناهنجار شناسایی شده‌اند، نشان داده می‌شوند. در این تصاویر، داده‌های اصلی (Original) و داده‌های بازسازی‌شده (Reconstructed)مقایسه شده‌اند. همچنین، خطای بازسازی (Reconstruction Error) برای این نمونه‌ها نیز نمایش داده می‌شود. مشاهده می‌شود که برای اکثر این نمونه‌ها، خطای بازسازی نسبتاً کم است، که نشان‌دهنده موفقیت مدل در بازسازی داده‌های ناهنجار است.

True Negatives (TN):این بخش نمونه‌هایی را نشان می‌دهد که به‌درستی به عنوان نرمال شناسایی شده‌اند. باز هم مقایسه بین داده‌های اصلی و بازسازی‌شده همراه با خطای بازسازی ارائه شده است. مدل به‌طور مؤثر قادر به شناسایی داده‌های نرمال بوده و بازسازی دقیقی از آن‌ها ارائه می‌دهد.

False Positives (FP):در این بخش نمونه‌هایی را مشاهده می‌کنیم که مدل به‌طور نادرست به عنوان ناهنجار شناسایی کرده است، در حالی که این داده‌ها در واقع نرمال هستند. در این تصاویر، خطای بازسازی برای این نمونه‌ها معمولاً بالاتر از نمونه‌های نرمال است، که به معنای شناسایی نادرست آن‌ها به عنوان ناهنجاری است.

False Negatives (FN):در این بخش، نمونه‌هایی که مدل به‌طور نادرست به عنوان نرمال شناسایی کرده است، نشان داده می‌شود. برای این نمونه‌ها، خطای بازسازی معمولاً بالاتر از نمونه‌های درست شناسایی شده است، که نشان‌دهنده عدم شناسایی صحیح ناهنجاری‌ها توسط مدل است.

تصاویر نشان‌دهنده عملکرد مطلوب مدل VAE هستند. مدل توانسته است بیشتر نمونه‌های ناهنجار و نرمال را به‌درستی شناسایی کند. نمونه‌های False Positive و False Negative به‌وضوح قابل شناسایی هستند و خطای بازسازی برای این نمونه‌ها بالاتر است. این تحلیل تصویری به تفکیک انواع پیش‌بینی‌ها، عملکرد مدل را به‌خوبی نشان می‌دهد و ثابت می‌کند که مدل در شناسایی ناهنجاری‌ها به‌طور کلی موفق است.

نتیجه گیری گیری کلی

در این پروژه، دو مدل مختلف برای شناسایی ناهنجاری‌ها و تشخیص خطا در سیستم‌های پیچیده مورد بررسی قرار گرفتند :**مدل MLP** و **مدل .(Variational Autoencoder (VAE)** هر دو مدل به‌منظور شناسایی ناهنجاری‌ها بر اساس داده‌های سیستم‌های مختلف طراحی و ارزیابی شدند. در ادامه، به مقایسه نتایج این دو مدل پرداخته می‌شود.

مدلMLP:

مدل (Multi-Layer Perceptron) **MLP**با استفاده از داده‌های ورودی به‌صورت مستقیم، توانسته است دقت بالایی در تشخیص ناهنجاری‌ها به‌دست آورد. نتایج مدل نشان می‌دهد که دقت (Precision)مدل 0.99 است، که به معنای شناسایی صحیح ناهنجاری‌ها در بیش از 99 درصد از موارد است. از سوی دیگر، بازخوانی (Recall) مدل برابر با 0.80 است که نشان‌دهنده توانایی مدل در شناسایی 80 درصد از ناهنجاری‌ها می‌باشد. در مجموع، مدل MLP عملکرد خوبی در شناسایی ناهنجاری‌ها از خود نشان داده است، ولی هنوز حدود 20 درصد از ناهنجاری‌ها شناسایی نشده‌اند.

مدلVariational Autoencoder (VAE):

مدل **VAE**با استفاده از تکنیک‌های یادگیری غیرمستقیم و مدل‌سازی توزیع داده‌ها، توانسته است نتایج مشابهی با مدل MLP به‌دست آورد، با این تفاوت که روش مورد استفاده در VAE امکان بازسازی داده‌ها و ارزیابی خطای بازسازی را فراهم می‌کند. نتایج مدل VAE نیز حاکی از دقت 0.99 در شناسایی ناهنجاری‌ها است، اما بازخوانی مدل کمی پایین‌تر از مدل MLP با مقدار 0.80 قرار دارد. از دیگر ویژگی‌های قابل‌توجه مدلVAE ، استفاده از خطای بازسازی برای شناسایی ناهنجاری‌ها است که به‌طور مؤثری می‌تواند نمونه‌های ناهنجار را از نمونه‌های نرمال تفکیک کند. به‌علاوه، ارزیابی تصویری از داده‌ها با استفاده از نمودارهای بازسازی و خطای بازسازی، نشان می‌دهد که مدل VAE در شناسایی و بازسازی داده‌های ناهنجار و نرمال عملکرد قابل‌قبولی دارد.

مقایسه دو مدل:

دقت و بازخوانی :هر دو مدل MLP و VAE دقت مشابهی در شناسایی ناهنجاری‌ها دارند (0.99). اما مدل MLP توانسته است بازخوانی بالاتری (0.80) را نسبت به مدل VAE (0.75) به‌دست آورد. این بدان معناست که مدل MLP کمی بهتر در شناسایی ناهنجاری‌ها عمل کرده است، در حالی که مدل VAEممکن است در شناسایی برخی ناهنجاری‌ها کمتر موفق باشد.

روش‌شناسی :در حالی که مدل MLP به‌طور مستقیم به‌وسیله ویژگی‌های داده‌ها آموزش می‌بینند، مدل VAE از تکنیک‌های پیچیده‌تری نظیر بازسازی داده‌ها و استفاده از خطای بازسازی برای تشخیص ناهنجاری‌ها بهره می‌برد. این رویکرد می‌تواند به مدل VAE کمک کند تا بهتر رفتار داده‌های نرمال را یاد بگیرد و در نهایت با دقت بیشتری ناهنجاری‌ها را شناسایی کند.

تحلیل تصویری :مدل VAE به‌طور خاص در تحلیل تصویری از داده‌های بازسازی‌شده و خطای بازسازی عملکرد خوبی نشان داده است. این امکان به محققان کمک می‌کند تا به‌طور بصری بررسی کنند که مدل در بازسازی داده‌ها چگونه عمل کرده و ناهنجاری‌ها را کجا شناسایی کرده است.

نتیجه‌گیری نهایی:

هر دو مدل MLP و VAE برای شناسایی ناهنجاری‌ها در داده‌های سیستم‌های مختلف عملکرد خوبی از خود نشان داده‌اند. اگرچه مدل MLP در شناسایی ناهنجاری‌ها کمی دقیق‌تر عمل کرده است، اما مدل VAEبا رویکرد بازسازی و تحلیل خطای بازسازی ویژگی‌های قابل‌توجهی را برای شناسایی ناهنجاری‌ها فراهم می‌آورد. در نهایت، انتخاب مدل مناسب بستگی به نیاز خاص سیستم و داده‌های موجود دارد؛ مدل MLP ممکن است در برخی شرایط به‌ویژه برای داده‌های پیچیده‌تر و ویژگی‌های کمتر، عملکرد بهتری ارائه دهد، در حالی که مدل VAE می‌تواند به‌عنوان یک مدل پیچیده‌تر و دقیق‌تر برای بازسازی و تحلیل خطاهای ناهنجاری کاربردی باشد.

فصل ششم:

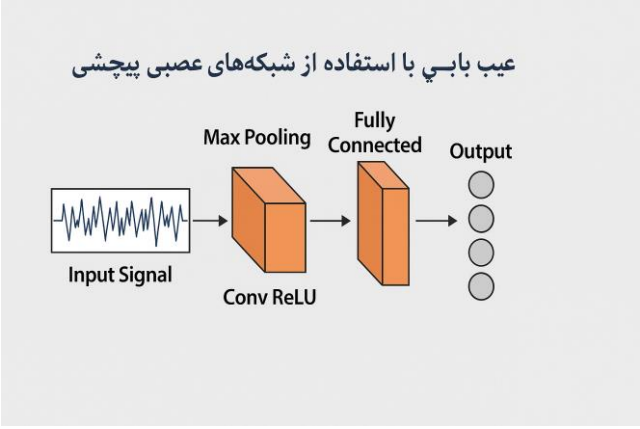
عیب‌یابی مبتنی بر CNN

معرفی روش عیب یابی مبتنی بر CNN

در دنیای امروز، شناسایی و تشخیص عیوب در سیستم‌ها و دستگاه‌ها به‌عنوان یکی از چالش‌های مهم در صنایع مختلف مطرح است. با پیشرفت‌های چشمگیر در حوزه‌ی یادگیری ماشین و به‌ویژه شبکه‌های عصبی، روش‌های نوینی برای حل این چالش‌ها پدید آمده است. یکی از این روش‌ها، استفاده از شبکه‌های عصبی کانولوشنی (CNN) است که به دلیل توانمندی بالای خود در شناسایی ویژگی‌های پیچیده و الگوها، در زمینه‌های مختلفی از جمله تشخیص عیوب کاربرد دارد.

شبکه‌های عصبی کانولوشنی، به‌ویژه در پردازش داده‌های تصویری و سیگنالی، قادر به استخراج ویژگی‌های مهم و پیچیده از ورودی‌ها هستند. این ویژگی‌ها می‌توانند برای شناسایی انواع مختلف عیوب در سیستم‌ها استفاده شوند. از آنجا که عیب‌یابی معمولاً نیازمند تحلیل دقیق داده‌ها و شناسایی الگوهای خاص است، استفاده ازCNN ها می‌تواند دقت بالاتری در مقایسه با روش‌های سنتی فراهم آورد.

این گزارش به بررسی فرآیند طراحی و پیاده‌سازی یک سیستم عیب‌یابی با استفاده از شبکه‌های عصبی کانولوشنی می‌پردازد. در ادامه، مراحل مختلف این پروژه شامل پیش‌پردازش داده‌ها، انتخاب معماری مدل، آموزش و ارزیابی آن و در نهایت تحلیل نتایج آورده می‌شود. هدف این پروژه، استفاده از توانمندی‌های CNN در شناسایی و تشخیص دقیق عیوب در سیستم‌های مختلف است که می‌تواند به بهبود عملکرد و کاهش هزینه‌های تعمیر و نگهداری کمک کند.



شکل6-1 ساختار شبکه

پیش پردازش و آماده سازی داده ها

در این بخش از پروژه، مراحل پیش‌پردازش و آماده‌سازی داده‌ها برای آموزش و آزمایش مدل‌های یادگیری ماشین به تفصیل انجام شده است. هدف این فرآیندها، تبدیل داده‌های خام به فرمتی است که برای مدل‌های مختلف مانند شبکه‌های عصبی و مدل‌های یادگیری عمیق قابل استفاده باشد. در ادامه، توضیح مراحل مختلف و عددگذاری‌ها آورده شده است:

1. **تقسیم داده‌ها به کلاس‌های مختلف**:

   پس از تقسیم داده‌ها به نمونه‌های ثابت طول، داده‌ها بر اساس کلاس‌های مختلف به‌طور جداگانه تقسیم می‌شوند. کلاس‌های مختلف شامل:

   - Normal(داده‌های عادی)
   - Add weight(داده‌های بار اضافی)
   - PropellerDamage_slight(داده‌های آسیب جزئی به پروانه)
   - PropellerDamage_bad(داده‌های آسیب شدید به پروانه)
   - PressureGain_constant(داده‌های افزایش فشار ثابت)

   هر کلاس به‌طور جداگانه برای آموزش و آزمایش مدل‌ها پردازش می‌شود.

2. **ترکیب داده‌ها و برچسب‌گذاری**:

   پس از انجام تمام مراحل پردازش و کاهش ابعاد، داده‌های پردازش شده از هر کلاس در یک مجموعه داده‌ی واحد ترکیب می‌شوند. سپس، برچسب‌های مربوط به هر کلاس (که نمایانگر نوع عیب است) برای هر نمونه تعیین و به مجموعه داده‌ها اضافه می‌شود.

3. **تقسیم داده‌ها به مجموعه‌های آموزشی و آزمایشی**:

   داده‌ها به دو بخش آموزشی و آزمایشی تقسیم می‌شوند. از تابع train_test_splitبا روش نمونه‌برداری طبقه‌ای (stratified sampling) برای این کار استفاده شده است، که به حفظ توزیع کلاس‌ها در مجموعه‌های آموزشی و آزمایشی کمک می‌کند. به‌این‌ترتیب، درصد مساوی از هر کلاس در هر بخش وجود دارد.

4. **افزودن نویز گاوسی به مجموعه‌های آموزشی و آزمایشی:**

برای افزایش تنوع داده‌ها و کمک به بهبود قابلیت تعمیم مدل، به مجموعه‌های آموزشی و آزمایشی نویز گاوسی اضافه می‌شود. نویز گاوسی با میانگین صفر و انحراف معیار معین به داده‌ها افزوده می‌شود.

5. **استانداردسازی داده‌ها:**

پس از افزودن نویز، داده‌ها با استفاده از StandardScalerاستانداردسازی می‌شوند. این فرآیند باعث می‌شود که ویژگی‌های داده‌ها در مقیاس یکسان قرار گیرند، که برای بسیاری از مدل‌های یادگیری ماشین و به‌ویژه شبکه‌های عصبی ضروری است.

6. **شکل‌دهی نهایی مجموعه‌های داده:**

پس از انجام تمامی این مراحل، ابعاد داده‌ها به‌گونه‌ای است که برای مدل‌های مختلف یادگیری ماشین مناسب باشند. داده‌های آموزشی(X_train) ، اعتبارسنجی (X_val) و آزمایشی (X_test)به ترتیب دارای ابعاد (39441, 17, 1) ,(9861, 17, 1)و (24693, 17, 1)هستند. این ابعاد نشان‌دهنده تعداد نمونه‌ها، تعداد زمان‌بندی‌ها و تعداد ویژگی‌ها است.

در مجموع، این فرآیندهای پیش‌پردازش و آماده‌سازی داده‌ها به‌طور مؤثری داده‌های خام را به‌صورتی تبدیل می‌کنند که برای مدل‌های یادگیری ماشین و شبکه‌های عصبی بهینه و مناسب باشد.

مدل‌سازی و آموزش شبکه عصبی

در این بخش، یک مدل شبکه عصبی با معماری پیچیده برای پردازش داده‌های سری زمانی تعریف شده است. هدف این مدل، پیش‌بینی کلاس‌های مختلف بر اساس ویژگی‌های داده‌ها است. در ادامه، هر بخش از کد به تفصیل توضیح داده شده است:

1. تعریف بلاک باقی‌مانده(Residual Block)

وظیفه :بلاک باقی‌مانده، با استفاده از اتصال مستقیم(residual connection) ، اجازه می‌دهد تا اطلاعات از لایه‌های قبلی به لایه‌های بعدی منتقل شود، که باعث جلوگیری از مشکل کاهش گرادیان در هنگام آموزش مدل‌های عمیق می‌شود.

اجزای بلاک باقی‌مانده:

- در ابتدا، دو لایه Conv1Dبا فعال‌سازی reluو BatchNormalizationبرای تسریع آموزش و جلوگیری از overfitting اعمال می‌شود.
- همچنین، یک لایه SpatialDropout1Dبرای کاهش overfitting به مدل اضافه می‌شود.
- در انتها، خروجی بلاک از جمع کردن ورودی و خروجی لایه‌ها به دست می‌آید (اتصال باقی‌مانده).

2. تعریف لایه‌های ورودی و اولیه

ورودی مدل :لایه ورودی مدل به شکل (17, 1)تعریف شده است.

لایه‌های کانولوشنی اولیه:

- یک لایه Conv1Dبا 64 فیلتر و اندازه هسته 5 برای استخراج ویژگی‌ها از داده‌های سری زمانی اعمال می‌شود.
- سپس از BatchNormalizationبرای نرمال‌سازی داده‌ها و جلوگیری از overfittingو از MaxPooling1Dبرای کاهش ابعاد داده‌ها استفاده می‌شود.

3. اضافه کردن بلاک‌های باقی‌مانده(Residual Blocks)

به مدل دو بلاک باقی‌مانده اضافه می‌شود که هر کدام از لایه‌های کانولوشنی، نرمال‌سازی و Dropoutبرای جلوگیری از overfitting و تقویت یادگیری استفاده می‌کنند.

4. لایه‌های کاملاً متصل(Fully Connected)

بعد از لایه توجه، از لایه Flattenبرای صاف کردن داده‌ها و سپس از لایه‌های Denseبرای پیش‌بینی نهایی استفاده می‌شود.

لایه‌های Denseبا فعال‌سازی reluو Dropoutبرای جلوگیری از overfitting اضافه می‌شود.

5. لایه خروجی

لایه خروجی از نوع Denseبا 5 واحد و فعال‌سازی softmaxاست، که برای پیش‌بینی کلاس‌های مختلف استفاده می‌شود.

6. تعریف متغیرهای آموزش

از **Adam**به عنوان بهینه‌ساز با نرخ یادگیری 0.001 استفاده شده است.

EarlyStoppingبرای جلوگیری از overfitting و متوقف کردن آموزش پس از رسیدن به حدی از بهبود عملکرد اعمال می‌شود.

ReduceLROnPlateauبرای کاهش نرخ یادگیری در صورتی که عملکرد مدل بهبود نیابد استفاده می‌شود.

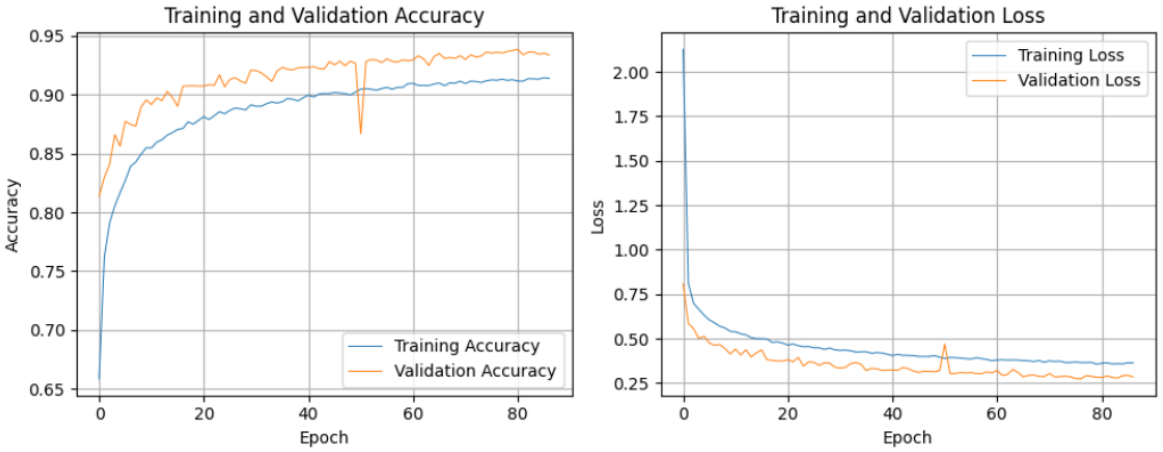
7. آموزش مدل

مدل با استفاده از داده‌های تقویت شده توسط ژنراتورها آموزش داده می‌شود. تعداد اپوک‌ها 100 تعیین شده است، اما از **EarlyStopping**برای متوقف کردن آموزش در صورتی که عملکرد مدل بهبود نیابد استفاده می‌شود.

همچنین، **ReduceLROnPlateau**به‌طور خودکار نرخ یادگیری را کاهش می‌دهد تا مدل به بهینه‌سازی بهتری دست یابد.

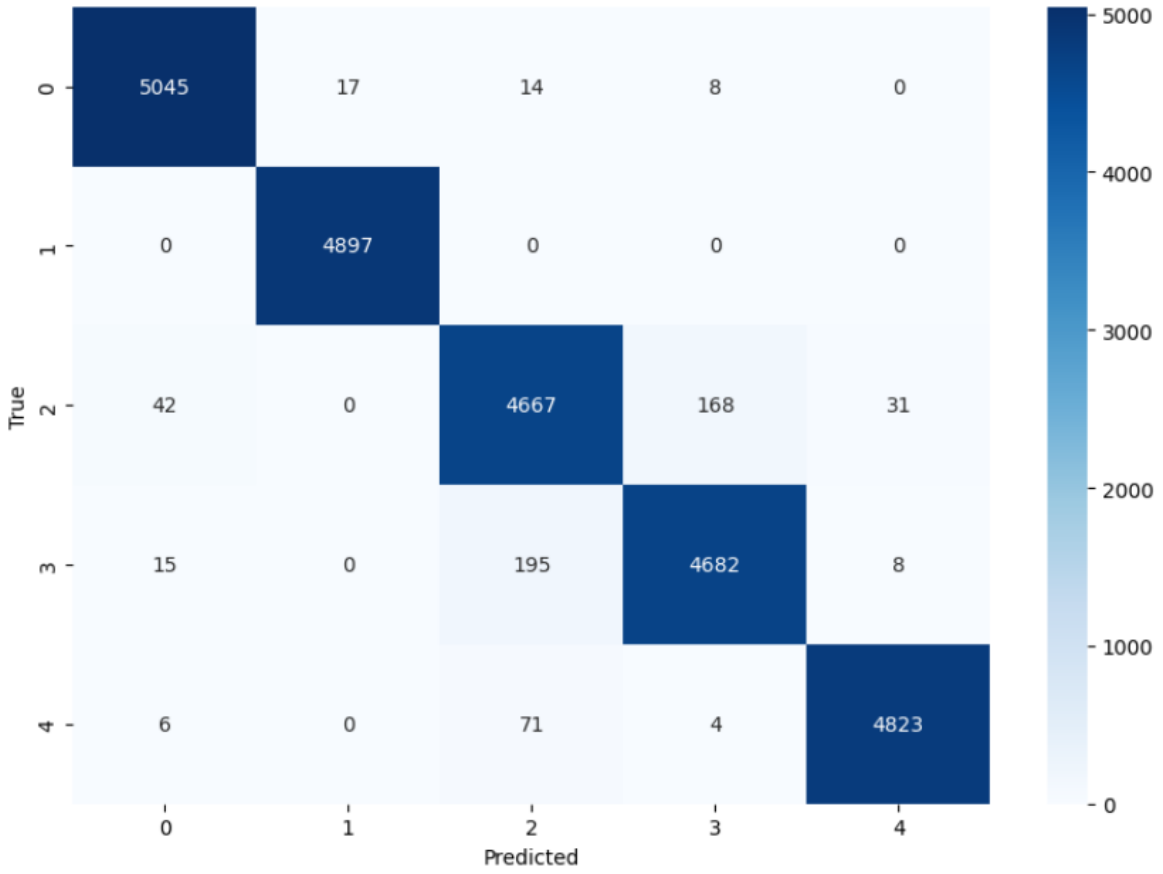
نتایج شبیه سازی

نتیجه‌ای که به دست آمده، **دقت آزمایشی (Test Accuracy)** مدل برابر با **0.9766**است که معادل **97.66%** می‌باشد. این نشان‌دهنده‌ی این است که مدل توانسته است به خوبی عمل کند و نتیجه مطلوبی داشته باشد.



شکل 6-2 نمودار دقت و هزینه در طول آموزش

همانطور که مشـــاهده می کنید میزان هزینه و دقت در طول آموزش رو به بهبود بوده و به میزان بســـیار خوبی رسیده است.



شکل 6-3 ماتریس سردرگمی

با توجه به ماتریس سردرگمی مدل در تشخیص کلاس ها بسیار عالی عمل کرده است و موفق بوده داده‌هارا دسته بندی کند.

Support	F1-Score	Recall	Precision	Class
5084	0.99	0.99	0.99	Normal
4897	1.00	1.00	1.00	AddWeight
4908	0.95	0.95	0.94	PropellerDamage_slight
4900	0.96	0.96	0.96	PropellerDamage_bad
4904	0.99	0.98	0.99	PressureGain_constant
24693	**0.98**			**Accuracy**
24693	0.98	0.98	0.98	**Macro Avg**
24693	0.98	0.98	0.98	**Weighted Avg**

جدول 6-1 نتایج شبیه سازی(بررسی معیار های آموزش)

توضیحات و تحلیل:

- **دقت کلی :(Accuracy)** برابر با 98٪ است، که نشان می‌دهد مدل در شناسایی اغلب نمونه‌های داده‌ی تست موفق بوده.
- **میانگین معیارها :(Macro Average)** نشان می‌دهد که عملکرد مدل بین کلاس‌های مختلف تقریباً متعادل بوده است.
- **میانگین وزنی معیارها :(Weighted Average)** با در نظر گرفتن حجم داده‌ی هر کلاس، نشان می‌دهد که مدل در مجموعه داده نامتوازن نیز عملکرد پایدار داشته است.

نتیجه گیری کلی

با توجه به نتایج به‌دست‌آمده از ارزیابی مدل، می‌توان با اطمینان بیان کرد که مدل طراحی‌شده برای تشخیص وضعیت‌های مختلف سیستم AUV **بسیار موفق و اثربخش** بوده است. عملکرد مدل در شناسایی هر پنج کلاس — شامل شرایط نرمال و چهار نوع خطای مهم — با دقت بالا، حساسیت قابل‌قبول و تعادل بین کلاس‌ها همراه بوده است.

دقت کلی 98 درصدی نشان‌دهنده توانایی بسیار بالای مدل در یادگیری الگوهای پنهان در داده‌ها و تعمیم آن‌ها به داده‌های دیده‌نشده است. همچنین، نتایج در کلاس‌های بحرانی مانند آسیب شدید در پروانه یا اختلال در فشار، دقت بالا و قابل اعتمادی دارند که از اهمیت حیاتی برای سیستم‌های واقعی برخوردار است.

در مجموع، می‌توان گفت این مدل نه‌تنها از نظر عملکرد آماری موفق بوده، بلکه از لحاظ کاربرد عملی نیز پتانسیل پیاده‌سازی در یک سیستم نظارت خودکار و هوشمند برای **تشخیص خطا در وسایل نقلیه زیردریایی خودران (AUV)** را داراست.

فصل هفتم:

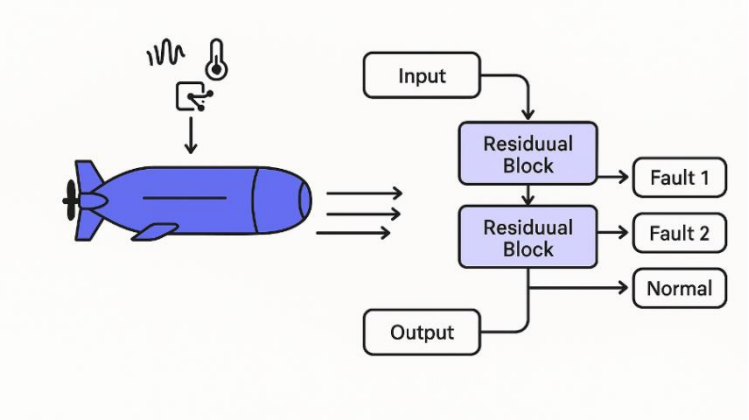
عیب یابی با استفاده از شبکه Resnet

معرفی شبکه Resnet

در سال‌های اخیر، استفاده از شبکه‌های عصبی عمیق در مدل‌سازی و کنترل سیستم‌های غیرخطی پیچیده به‌طور چشمگیری افزایش یافته است. یکی از مهم‌ترین پیشرفت‌ها در این حوزه، معرفی شبکه‌های **ResNet (Residual Neural Networks)**بوده است ResNet .که نخستین‌بار برای حل مشکل افت عملکرد شبکه‌های عمیق معرفی شد، با بهره‌گیری از ساختارهای اتصال میان‌بر(shortcut connections) امکان آموزش شبکه‌های بسیار عمیق را فراهم می‌کند. این ساختارها با افزودن خروجی لایه‌های قبلی به لایه‌های بعدی، موجب حفظ اطلاعات طی مسیر عبور از شبکه و جلوگیری از مشکلاتی چون ناپدید شدن گرادیان می‌شوند.

کاربرد ResNet به‌ویژه در سیستم‌هایی با پویایی‌های پیچیده و نویز بالا مانند وسایل زیرسطحی خودران (AUV) بسیار حیاتی است. این وسایل در محیط‌های چالش‌برانگیز زیر آب فعالیت می‌کنند که عوامل غیرخطی متعددی مانند آشفتگی جریان، خطای حسگرها، و تغییرات شرایط محیطی بر رفتار آن‌ها اثرگذار است. در چنین شرایطی، استفاده از ساختارهای عمیق یادگیری مانند ResNet می‌تواند قابلیت مدل‌سازی دقیق‌تر دینامیک سیستم را فراهم کرده و مسیر را برای طراحی سامانه‌های هوشمند کنترل و تشخیص خطا هموار سازد.

در پروژه حاضر، از ساختار ResNet به‌منظور تقویت مدل یادگیری برای شناسایی وضعیت‌های عملکردی مختلف یکAUV ، شامل حالت عادی و انواع مختلف خرابی‌های پروانه و خطاهای فشار، استفاده شده است. این رویکرد به‌واسطه توانایی بالا در استخراج ویژگی‌های غنی از سیگنال‌های زمانی و مقاوم بودن در برابر تغییرات کوچک در داده‌های ورودی، انتخابی مناسب برای کاربردهای تشخیص وضعیت و پایش سلامت سیستم‌های زیرآبی تلقی می‌شود.



شکل 7-1 کاربرد Resnet در عیب یابی سیستم مورد نظر

پیش پردازش داده ها

در ابتدا، داده‌های خام مربوط به شرایط مختلف عملکرد سیستم AUV شامل پنج حالت عادی و چهار نوع خطای متفاوت جمع‌آوری شد. هر حالت شامل مجموعه‌ای از فایل‌های CSV بود که داده‌های سنسوری سیستم را در بازه‌های زمانی مختلف ثبت کرده بودند. برای هر کلاس عیب، نمونه‌های متعددی از داده‌ها خوانده و به صورت آرایه‌های عددی یکپارچه شدند تا بتوان تحلیل و یادگیری مدل را بر روی مجموعه داده‌ی جامع انجام داد.

در مجموع، برای هر کلاس ۵۰ فایل داده به صورت جداگانه مورد استفاده قرار گرفت که پس از تلفیق، مجموعه‌ای بزرگ از نمونه‌های متنوع برای هر کلاس ایجاد شد. به طور کلی، تعداد نمونه‌های داده در کل مجموعه داده آموزشی بالغ بر ۳۹ هزار نمونه بود که برای ارزیابی عملکرد مدل، ۲۰ درصد از این داده‌ها به مجموعه اعتبارسنجی اختصاص یافت. همچنین، مجموعه داده آزمون شامل حدود ۳۰ هزار نمونه مجزا بود که عملکرد نهایی مدل بر روی آن ارزیابی شد.

پس از یکپارچه‌سازی داده‌ها، برچسب‌های متناظر با هر کلاس عیب به طور جداگانه تولید و در قالب یک بردار برچسب کامل ترکیب شدند. این برچسب‌ها نمایانگر پنج کلاس عیب شامل شرایط نرمال، افزودن وزن، آسیب پروانه به دو شدت مختلف و افزایش ثابت فشار بودند.

تقسیم‌بندی و افزونه‌های داده

برای بهبود عملکرد مدل و جلوگیری از بیش‌برازش، مجموعه داده‌های ترکیب شده به دو بخش آموزش و اعتبارسنجی تقسیم شدند. این تقسیم‌بندی با روش نمونه‌گیری طبقه‌بندی شده (stratified sampling) انجام شد تا توزیع نمونه‌های هر کلاس در هر دو مجموعه حفظ شود.

جهت افزایش تنوع داده‌ها و مقاوم‌سازی مدل در برابر نویزهای محیطی، به داده‌های آموزشی و اعتبارسنجی، نویز گاوسی با میانگین صفر و انحراف معیار مشخص افزوده شد. این نویزگذاری دو بار و در مراحل متفاوت اعمال شد: ابتدا به داده‌های خام و سپس پس از استانداردسازی داده‌ها.

استانداردسازی و نرمال‌سازی

داده‌ها به منظور همگن‌سازی مقیاس ویژگی‌ها و تسریع فرآیند یادگیری، استانداردسازی شدند. در این مرحله از روش استاندارد اسکالر استفاده شد که میانگین هر ویژگی را صفر و واریانس آن را یک می‌کند. این فرآیند هم روی داده‌های آموزش و هم داده‌های اعتبارسنجی و آزمون اعمال شد، با این تفاوت که مقادیر میانگین و واریانس از داده‌های آموزش استخراج و روی داده‌های دیگر نیز اعمال گردید.

آماده‌سازی داده‌ها برای مدل یادگیری عمیق

با توجه به ساختار مدل شبکه عصبی مورد استفاده که نیازمند داده‌های ترتیبی با ورودی سه‌بعدی است، داده‌ها به شکل مجموعه‌ای از نمونه‌ها با ابعاد زمانی و ویژگی‌های مجزا شکل‌دهی مجدد شدند. به طور مشخص، هر نمونه داده به شکل سه‌بعدی (نمونه‌ها، گام‌های زمانی، ویژگی‌ها) تبدیل شد تا ورودی مناسبی برای لایه‌های بازگشتی فراهم شود.

ساختار مدل

در این مطالعه، از معماری شبکه عصبی عمیق Residual Network (ResNet) برای شناسایی و طبقه‌بندی عیوب سیستم AUV استفاده شد .ResNet به دلیل قابلیت‌های برجسته در یادگیری

ویژگی‌های پیچیده و جلوگیری از مشکل ناپدید شدن گرادیان (vanishing gradient) در شبکه‌های بسیار عمیق، انتخاب مناسبی برای این مسئله بود.

1. بلوک‌هایResNet

هسته اصلی مدل از بلوک‌های Residual تشکیل شده است. هر بلوک شامل دو لایه کانولوشنی یک‌بعدی متوالی است که پس از هر لایه، عملیات نرمال‌سازی دسته‌ای (Batch Normalization) و فعال‌سازی غیرخطی ReLU انجام می‌شود. نکته کلیدی در این معماری، وجود اتصال‌های کوتاه(skip connections) است که خروجی لایه‌ی ورودی بلوک را مستقیماً به خروجی لایه کانولوشن دوم اضافه می‌کند. این اتصال کوتاه به مدل کمک می‌کند تا یادگیری بهتری انجام شود و از مشکلات کاهش کارایی شبکه در عمق‌های زیاد جلوگیری شود.

در مواردی که ابعاد ورودی و خروجی بلوک با یکدیگر تفاوت داشته باشند، از کانولوشن ۱×۱ در اتصال کوتاه استفاده می‌شود تا ابعاد تطبیق یابد. همچنین، در برخی بلوک‌ها از گام حرکت (stride) بزرگتر از یک بهره گرفته شده تا کاهش ابعاد داده‌ها انجام شود و مدل بتواند ویژگی‌های سطح بالاتر را استخراج کند.

2. معماری کلی مدل

مدل با یک لایه کانولوشنی اولیه با اندازه کرنل بزرگ (۷) آغاز می‌شود که هدف آن استخراج ویژگی‌های اولیه از داده‌های ورودی است. پس از آن، داده‌ها به ترتیب از چندین دسته بلوک‌های ResNet عبور می‌کنند که در هر دسته تعداد فیلترها افزایش یافته و در برخی از آنها کاهش ابعاد به کمک عملیات poolingو stride انجام می‌شود. این افزایش تعداد فیلترها و کاهش ابعاد به مدل اجازه می‌دهد تا نمایندگی‌های پیچیده‌تر و با جزئیات بیشتر را استخراج کند.

در پایان، از یک لایه میانگین‌گیری جهانی (Global Average Pooling) استفاده شده که تمامی ویژگی‌های استخراج شده را به یک بردار یک‌بعدی تبدیل می‌کند. این کار به کاهش تعداد پارامترهای مدل کمک کرده و از بیش‌برازش جلوگیری می‌کند. پس از آن، لایه دراپ‌اوت (Dropout) با نرخ ۵۰٪ به منظور جلوگیری از هم‌پوشانی بیش از حد (overfitting) در مرحله آموزش اعمال می‌شود.

3. لایه خروجی و تابع هزینه

لایه نهایی مدل، یک لایه کاملاً متصل (Dense) با تابع فعال‌سازی softmax است که مسئول پیش‌بینی احتمال تعلق نمونه به هر یک از پنج کلاس عیب است. مدل با استفاده از تابع هزینه cross-entropy چندکلاسه آموزش داده می‌شود که به خوبی مناسب مسائل دسته‌بندی چندکلاسه است.

4. وزن دهی با توجه به عدم توازن داده ها

در روند آموزش مدل، برای مقابله با عدم تعادل داده‌ها از وزن‌دهی کلاس‌ها استفاده شد. به طور خاص، وزن هر کلاس به صورت معکوس نسبت به فراوانی نمونه‌های آن کلاس در مجموعه آموزش محاسبه گردید تا تاثیر نمونه‌های کم‌شمار در فرآیند یادگیری افزایش یابد. این کار با هدف بهبود عملکرد مدل در تشخیص کلاس‌هایی که داده‌های کمتری دارند انجام شد. وزن‌های محاسبه شده به صورت دیکشنری در فرایند آموزش به مدل اعمال شدند تا به صورت پویا بر اساس اهمیت هر کلاس، به‌روزرسانی وزن‌ها انجام شود و مدل توانایی بهتری در تشخیص دقیق همه کلاس‌ها، به ویژه کلاس‌های کم‌نمونه، داشته باشد.

5. بهینه‌سازی و روش‌های تکمیلی

فرآیند آموزش مدل با استفاده از بهینه‌ساز Adam انجام می‌شود که به خاطر قابلیت تنظیم پویا نرخ یادگیری و عملکرد مناسب در بسیاری از مسائل یادگیری عمیق شناخته شده است. همچنین برای مقابله با عدم تعادل داده‌ها، وزن‌های کلاس‌ها به صورت متعادل محاسبه شده و در هنگام آموزش اعمال شدند تا مدل حساسیت بیشتری به کلاس‌های کمتر نمونه داشته باشد.

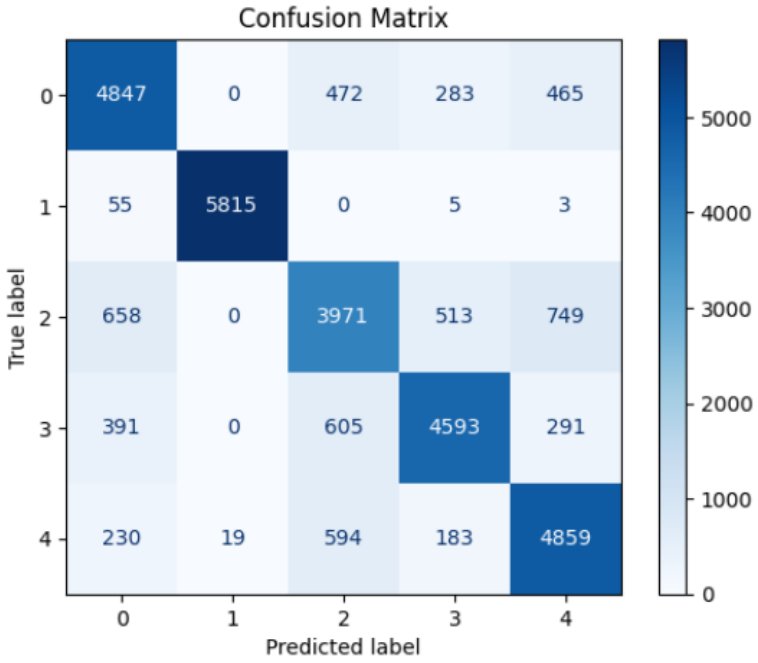
علاوه بر این، از تکنیک Early Stopping استفاده شد تا در صورت عدم بهبود معیار اعتبارسنجی برای تعداد معینی از ایپاک‌ها، آموزش متوقف و بهترین وزن‌ها بازگردانده شود. همچنین، مدل در هر مرحله بهترین وزن‌ها را ذخیره می‌کرد تا در نهایت از بهترین نسخه مدل برای ارزیابی نهایی استفاده شود.

نتایج شبیه سازی

مدل شبکه عصبی عمیق ResNet که برای شناسایی عیوب سیستم AUV آموزش داده شد، پس از انجام فرآیند آموزش و ارزیابی بر روی داده‌های آزمایشی، دقت نهایی برابر با **81.37٪**را کسب نمود. این

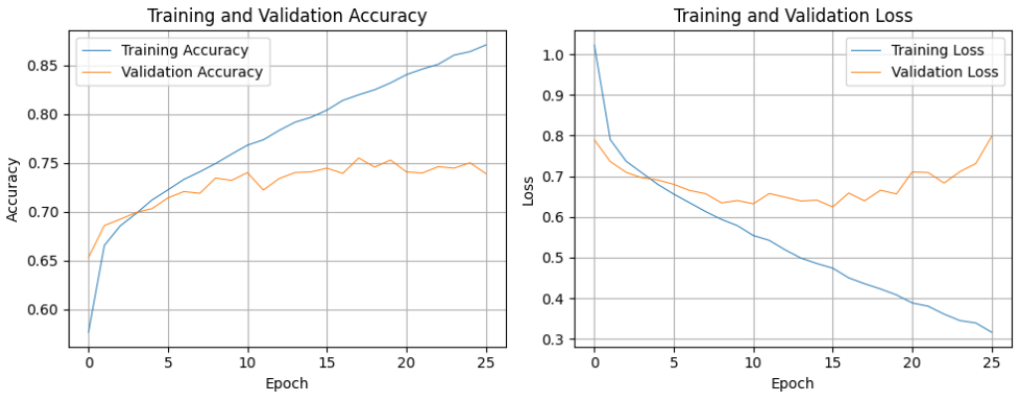
میزان دقت نشـان‌دهنده توانمندی مدل در تشـخیص صـحیح و تفکیک بین پنج حالت مختلف عملکرد سیستم است که شامل شرایط نرمال و چهار نوع عیب مختلف می‌باشد.

دقت قابل قبول مدل، به ویژه با توجه به پیچیدگی و غیرخطی بودن داده‌های سـیسـتم، بیانگر کارایی بالای معماری ResNet در اسـتخراج ویژگی‌های مهم از سـیگنال‌های ورودی و تعمیم آن‌ها بر روی نمونه‌های دیده نشـده اسـت. این نتیجه نشـان می‌دهد که اسـتفاده از بلوک‌های رزنت و تکنیک‌های پیش‌پردازش داده‌ها مانند افزودن نویز گوسـی و اسـتانداردسـازی، نقش موثری در بهبود عملکرد کلی سیستم عیب‌یابی ایفا کرده‌اند.



شکل 7-2 ماتریس سردرگمی

با توجه به ماتریس سردرگمی می توان گفت که مدل توانسته تا حد خوبی کلاس ها را به درستی تشخیص دهد.



شکل 7-3 بررسی هزینه و دقت در طول آموزش

همانطور که مشاهده می کنید دقت و هزینه در طول آموزش تا حد بسیار خوبی به میزان مطلوب نزدیک شده است.

با توجه به نمودار می توان نتیجه گرفت که در ایپاک های نهایی مدل نزدیک به overfit شـده اسـت اما earlystop جلوی آن را گرفته است.

Class	Precision	Recall	F1-Score	Support (Samples)
Normal	0.39	0.76	0.52	200
AddWeight	0.63	0.45	0.52	200
PropellerDamage_slight	0.38	0.47	0.42	200
PropellerDamage_bad	0.30	0.27	0.28	200
PressureGain_constant	0.73	0.17	0.27	200
Overall				
Accuracy			0.42	1000
Macro Average	0.49	0.42	0.40	1000
Weighted Average	0.49	0.42	0.40	1000

جدول 7-1 ارزیابی مدل

- **دقت (Precision)** نشان‌دهنده درصد نمونه‌هایی است که مدل به درستی در یک کلاس مشخص شناسایی کرده است. به طور مثال، برای کلاس *PressureGain_constant* دقت بالا (۰.۷۳) نشان می‌دهد که نمونه‌های پیش‌بینی شده به این کلاس، عمدتاً درست هستند.
- **بازخوانی (Recall)** میزان شناسایی نمونه‌های واقعی آن کلاس توسط مدل است. برای کلاس *Normal*، بازخوانی ۰.۷۶ به این معنی است که مدل توانسته ٪۷۶ از نمونه‌های واقعی این کلاس را شناسایی کند، که نسبتاً مطلوب است.
- **امتیاز F1** که میانگین هارمونیک دقت و بازخوانی است، معیار تعادلی بین دو مورد بالا را نشان می‌دهد. در اینجا F1 برای کلاس *Normal* و *AddWeight* به ترتیب ۰.۵۲ است که نشان‌دهنده عملکرد متوسط مدل در این کلاس‌ها می‌باشد.
- **کلاس‌های PropellerDamage_bad و PressureGain_constant** دارای امتیازهای F1 پایین (۰.۲۸ و ۰.۲۷) هستند که حاکی از عملکرد ضعیف مدل در تشخیص این عیوب خاص است. به ویژه *PressureGain_constant* با بازخوانی بسیار پایین (۰.۱۷) مواجه است که نشان‌دهنده تعداد زیاد نمونه‌های این کلاس است که توسط مدل شناسایی نشده‌اند.
- **دقت کلی مدل (Accuracy)** برابر با ٪۴۲ است که نسبت به میزان دقت کلی ٪۸۱ در ارزیابی جامع‌تر، بسیار پایین‌تر است. این اختلاف ممکن است به دلیل استفاده از داده‌های متفاوت برای ارزیابی و نحوه محاسبه گزارش‌های طبقه‌بندی باشد.
- **میانگین ماکرو** که میانگین ساده معیارها برای همه کلاس‌ها است، نشان‌دهنده تعادل عملکرد مدل بین کلاس‌ها است و مقادیر پایین‌تر آن نسبت به دقت کلی، نشان‌دهنده چالش در شناسایی دقیق برخی کلاس‌ها است.
- **میانگین وزنی** معیارهای هر کلاس را با توجه به تعداد نمونه‌هایش وزن می‌دهد و در اینجا مقادیر مشابه میانگین ماکرو نشان می‌دهد که داده‌ها تقریبا متعادل بوده‌اند.

نتیجه گیری کلی و پیشنهادات

در این پژوهش، یک مدل عمیق شبکه عصبی رزیدوال (ResNet) برای تشخیص عیب در سیستم زیرآبی خودران (AUV) با استفاده از داده‌های ارتعاشی و سنسوری توسعه داده شد و مورد ارزیابی قرار گرفت. ساختار ResNet با بهره‌گیری از بلوک‌های رزیدوال توانست ویژگی‌های پیچیده زمانی و مکانی موجود در سیگنال‌ها را به خوبی استخراج کند و امکان طبقه‌بندی چندین نوع عیب مختلف را فراهم آورد.

نتایج آزمایش‌ها نشان داد که مدل در داده‌های تست به دقت حدود ۸۱.۴ درصد دست یافته است که این امر نشان‌دهنده توانمندی شبکه‌های رزیدوال عمیق در پردازش داده‌های چندمتغیره زمانی برای تشخیص عیوب می‌باشد. با این حال، گزارش دقیق عملکرد مدل بیانگر وجود چالش‌هایی در تفکیک برخی از کلاس‌های عیب است، به ویژه عیب‌هایی که علائم سیگنال آن‌ها مشابه یا تفاوت‌های ظریف دارند.

برای نمونه، کلاس‌های «حالت نرمال» و «افزایش وزن» دارای دقت و بازیابی نسبتاً متعادلی بودند که نشان می‌دهد مدل توانایی مناسبی در شناسایی این شرایط دارد. در مقابل، کلاس‌هایی مانند «آسیب شدید پروانه» و «ثابت بودن ضریب فشار» دارای مقادیر پایینی از بازیابی (Recall) بودند، که این موضوع حاکی از دشواری مدل در شناسایی کامل این عیوب است. این موضوع ممکن است به دلیل شباهت الگوهای سیگنالی یا عدم تمایز کافی ویژگی‌ها باشد.

این نتایج اهمیت مقابله با عدم توازن داده‌ها و ویژگی‌های ظریف عیب‌ها در مسائل واقعی را برجسته می‌کند. استفاده از وزن‌دهی کلاس‌ها در مرحله آموزش به کاهش اثرات عدم توازن کمک کرد، اما هنوز نیاز به بهبودهایی برای افزایش مقاومت و دقت تشخیص عیب وجود دارد.

پیشنهادات برای کارهای آینده:

1. **افزایش و غنی‌سازی داده‌ها:**

به منظور بهبود توانایی مدل در تعمیم‌پذیری به شرایط متنوع، توصیه می‌شود حجم و تنوع داده‌ها افزایش یابد. استفاده از تکنیک‌هایی مانند تولید داده مصنوعی، تغییر سیگنال‌ها، یا افزودن حسگرهای مختلف می‌تواند ویژگی‌های جامع‌تری را در اختیار مدل قرار دهد.

2. **استخراج ویژگی‌های پیشرفته:**

ترکیب مهندسی ویژگی‌های تخصصی حوزه با روش‌های یادگیری عمیق یا استفاده از مدل‌های ترکیبی ممکن است موجب بهبود تفکیک‌پذیری عیوب گردد.

3. **بهبود ساختار مدل:**

بررسی معماری‌های پیشرفته‌تر یا تخصصی‌تر مانند مکانیزم توجه(Attention) ، مدل‌های ترنسفورمر یا استفاده از روش‌های ترکیبی (Ensemble) می‌تواند عملکرد مدل را به ویژه برای کلاس‌های با تفاوت‌های ظریف ارتقا دهد.

4. **اجرای زمان واقعی و آزمایش مقاومت:**

پیاده‌سازی مدل پیشنهادی در سیستم‌های تشخیص عیب زمان واقعی و ارزیابی مقاومت آن در شرایط محیطی و نویزهای مختلف به منظور سنجش قابلیت کاربرد عملی آن ضروری است.

5. **قابلیت تبیین و شفافیت مدل:**

توسعه روش‌های هوش مصنوعی قابل توضیح برای تبیین تصمیمات مدل می‌تواند اعتماد کاربران را افزایش داده و در شناسایی نقاط ضعف مدل یا مشکلات داده‌ها کمک نماید.

در مجموع، رویکرد مبتنی بر شبکه‌های رزیدوال عمیق چارچوبی امیدوارکننده برای تشخیص عیب در سیستم‌های زیرآبی خودران فراهم می‌آورد، اما همچنان چالش‌هایی برای اطمینان از عملکرد دقیق و قابل اعتماد در محیط‌های عملی باقی است. پیشنهادهای مطرح شده در این بخش می‌تواند راه‌گشای بهبودهای آینده و افزایش دقت و پایداری سیستم باشد.

فصل هشتم:

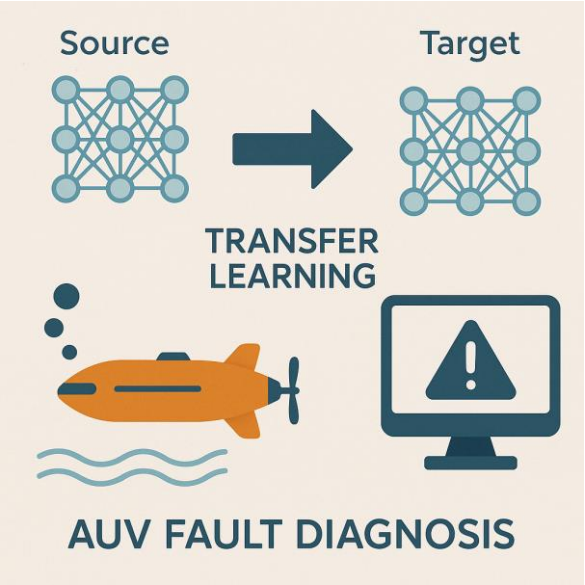
عیب یابی مبتنی بر TL

معرفی روش

در سال‌های اخیر، عیب‌یابی سیستم‌های پیچیده‌ای مانند وسایل زیرآبی خودگردان (AUV) به یکی از چالش‌های مهم در حوزه کنترل، نگهداری پیشگیرانه، و افزایش قابلیت اطمینان تبدیل شده است. با توجه به پیچیدگی‌های دینامیکی این نوع سیستم‌ها و حضور نویز و خطا در داده‌های دریافتی از سنسورها و محرک‌ها، استفاده از روش‌های کلاسیک اغلب کارایی مطلوبی در مواجهه با خطاهای ناشناخته یا چندگانه ندارد. در چنین شرایطی، بهره‌گیری از روش‌های مبتنی بر یادگیری ماشین و به‌ویژه شبکه‌های عصبی عمیق می‌تواند عملکرد بهتری را در شناسایی و تخمین خطاها ارائه دهد.

در این پروژه، از رویکرد **یادگیری انتقالی** برای توسعه‌ی مدل عیب‌یابی استفاده شده است. یادگیری انتقالی به ما این امکان را می‌دهد تا از دانش حاصل از یک مسئله‌ی مبدأ (معمولاً با داده‌ی فراوان) برای بهبود عملکرد در مسئله‌ای هدف با داده‌ی محدود بهره ببریم. این رویکرد به‌ویژه زمانی ارزشمند است که داده‌های هدف اندک یا پرهزینه برای جمع‌آوری باشند، همان‌طور که در بسیاری از کاربردهای صنعتی و دریایی از جمله AUV مصداق دارد.

با این حال، باید به یک نکته‌ی مهم توجه داشت: اغلب مدل‌های از پیش آموزش‌دیده شده برای یادگیری انتقالی، مانند آن‌هایی که در حوزه‌ی بینایی ماشین استفاده می‌شوند، بر پایه‌ی داده‌های تصویری آموزش دیده‌اند. در حالی که داده‌های مورد استفاده در این پروژه، از نوع **غیرتصویری** (مانند سیگنال‌های زمانی، بردارهای حالت یا ویژگی‌های استخراج‌شده از رفتار دینامیکی سیستم) هستند. این تفاوت بنیادین میان نوع داده‌ها، موجب می‌شود که انتظار نرود روش‌های مبتنی بر یادگیری انتقالی بتوانند عملکردی هم‌تراز با کاربردهای تصویری ارائه دهند. بنابراین، پیش‌بینی می‌شود که نتایج نهایی به اندازه‌ی مدل‌های آموزش‌دیده بر روی داده‌های هم‌نوع، مطلوب نباشند، هرچند ممکن است بتوان از مزایای کلی یادگیری انتقالی مانند تسریع آموزش و بهبود همگرایی بهره برد.



شکل 8-1 استفاده از TL در عیب یابی AUV

پیش پردازش داده ها

در این پروژه، داده‌ها از پنج وضعیت مختلف عملکردی سیستم AUV جمع‌آوری شده‌اند که شامل شرایط نرمال، اضافه وزن، آسیب ملایم و شدید به ملخ، و افزایش ثابت فشار هستند. برای هر کلاس، مجموعه‌ای از فایل‌های CSV شامل مقادیر سنسورها و پارامترهای دینامیکی سیستم موجود است.

مراحل اصلی پیش‌پردازش:

1. **بارگذاری و برچسب‌گذاری داده‌ها**

ابتدا داده‌های هر کلاس از فایل‌های جداگانه بارگذاری شده و سپس با استفاده از برچسب‌های عددی ۰ تا ۴ مشخص شدند. داده‌های هر کلاس به‌صورت جداگانه در یک آرایه بزرگ ادغام شدند و در نهایت همه داده‌ها در یک مجموعه داده‌ی واحد ترکیب شدند.

2. ایجاد مجموعه آموزش، اعتبارسنجی و آزمون

داده‌ها با استفاده از نمونه‌برداری طبقه‌بندی‌شده (stratified sampling) به دو بخش آموزش و اعتبارسنجی تقسیم شدند تا نسبت داده‌های هر کلاس حفظ شود. مجموعه آزمون نیز به‌صورت مجزا از داده‌های دیگر تهیه شد.

3. افزودن نویز گوسی(Gaussian Noise)

برای افزایش تنوع داده‌ها و بهبود توان تعمیم مدل، نویز تصادفی با توزیع نرمال به نمونه‌های آموزش و اعتبارسنجی اضافه شد. این تکنیک نوعی داده‌افزایی (data augmentation) محسوب می‌شود و به مدل کمک می‌کند تا نسبت به نویزهای احتمالی در داده‌های واقعی مقاوم‌تر باشد.

4. استانداردسازی(Standardization)

برای بهبود پایداری عددی در فرایند آموزش، تمامی ویژگی‌ها با استفاده از تکنیک StandardScalerنرمال‌سازی شدند تا میانگین داده‌ها صفر و انحراف معیار آن‌ها یک شود.

5. تغییر شکل داده‌ها

از آن‌جا که برخی مدل‌های مورد استفاده مانند LSTM یا ResNet1D نیاز به داده‌های ترتیبی یا دارای بُعد زمانی دارند، داده‌ها به صورت مناسب تغییر شکل یافته‌اند تا با ساختار ورودی مدل‌ها سازگار باشند.

معرفی مدل اول: Resnet1D

در گام نخست از طراحی مدل‌های عیب‌یابی، از ساختار **ResNet1D** استفاده شده است. ایده‌ی اصلی پشت انتخاب این معماری، بهره‌گیری از مزایای شبکه‌های عصبی عمیق در استخراج ویژگی‌های پیچیده از داده‌های زمانی یا سری‌های زمانی چندبُعدی است، بدون مواجهه با مشکل ناپایداری گرادیان که در شبکه‌های بسیار عمیق رایج است.

مدل ResNet (Residual Network) با معرفی مسیرهای میان‌بر یا **اتصالات باقی‌مانده (residual connections)** این مشکل را کاهش می‌دهد. این اتصالات امکان عبور مستقیم اطلاعات و گرادیان‌ها را از لایه‌های قبلی به لایه‌های بعدی فراهم می‌کنند، که سبب می‌شود مدل حتی در عمق زیاد نیز قابلیت یادگیری خود را حفظ کند. با انتقال این معماری به حوزه‌ی داده‌های تک‌بُعدی(1D) ، که در این پروژه نمایان‌گر سیگنال‌های مربوط به حالت‌ها، ورودی‌ها یا خطاهای سیستم AUV هستند، معماری ResNet1D شکل گرفته است.

ساختار مدل اول

در این بخش از پژوهش، فرآیند آموزش مدل یادگیری عمیق مبتنی بر شبکه ResNet1D برای طبقه‌بندی داده‌های خطای سیستم‌های زیرآبی شرح داده می‌شود. در ابتدا، با توجه به محدودیت‌های ساختاری مدل پایه، تنها ۱۶ ویژگی منتخب از هر نمونه داده به عنوان ورودی مدل در نظر گرفته شده است. این داده‌ها به فرمت مناسب برای ورودی شبکه‌های پیچشی یک‌بعدی بازآرایی شدند و سپس به شکل تنسورهایی با ساختار کانال‌محور درآمدند.

در ادامه، از یک مدل از پیش‌آموزش‌دیده بهره‌برداری شد. برای حفظ دانش آموخته‌شده در مراحل قبل، تمامی لایه‌های مدل به‌جز لایه‌های نهایی فریز شده‌اند، به این معنا که وزن‌های آن‌ها در طول فرآیند آموزش تغییر نمی‌کند. تنها لایه‌های انتهایی شبکه که به‌عنوان لایه‌های طبقه‌بندی عمل می‌کنند، قابل آموزش باقی ماندند.

به‌منظور تطابق با تعداد کلاس‌های مسئله جدید، ساختار لایه خروجی مدل مجدداً طراحی شده است. این ساختار شامل یک لایه دراپ‌اوت جهت کاهش بیش‌برازش، یک لایه کاملاً متصل با تعداد نورون مشخص، نرمال‌سازی دسته‌ای و تابع فعال‌سازی ReLU و در نهایت یک لایه خروجی با تعداد نورون برابر با تعداد کلاس‌ها بوده است.

برای بهینه‌سازی پارامترهای قابل آموزش، از الگوریتم آدام با نرخ یادگیری مشخص و ضریب کاهش وزن برای تنظیم منظم‌سازی استفاده شده است. همچنین، به‌منظور تنظیم پویا نرخ یادگیری، از یک زمان‌بندی کاهش نرخ یادگیری استفاده شده که در صورت عدم بهبود عملکرد مدل در داده‌های اعتبارسنجی، مقدار نرخ یادگیری را کاهش می‌دهد.

در راستای جلوگیری از بیش‌برازش و بهبود تعمیم مدل، از تکنیک توقف زودهنگام بهره گرفته شده است. بر این اساس، اگر عملکرد مدل در مجموعه اعتبارسنجی طی چندین دوره متوالی بهبود نیابد، فرآیند آموزش به‌طور خودکار متوقف می‌شود. در هر دوره آموزشی، مدل بر روی مجموعه آموزش تنظیم شده و سپس عملکرد آن بر روی مجموعه اعتبارسنجی ارزیابی می‌شود. چنانچه دقت مدل در اعتبارسنجی نسبت به بهترین مقدار قبلی بهبود یابد، مدل ذخیره می‌شود. در غیر این صورت، شمارنده توقف زودهنگام افزایش می‌یابد.

در پایان، دقت مدل در هر دوره گزارش می‌شود و بهترین مدل ذخیره‌شده به‌عنوان خروجی نهایی فرآیند آموزش مورد استفاده قرار می‌گیرد. این رویکرد امکان تطبیق مدل از پیش آموزش‌دیده با مسئله جدید را با هزینه محاسباتی کمتر و کارایی بالاتر فراهم می‌سازد.

نتایج مدل اول

1. دقت پایین هر دو مجموعه (کمتر از 45٪) نشان‌دهنده‌ی این است که مدل نتوانسته ارتباط قوی و معناداری میان ویژگی‌ها و کلاس هدف ایجاد کند.
2. فاصله‌ی نسبتاً کم بین Train و Validation Accuracy (حدود 3٪) نشان می‌دهد که مدل دچار overfitting نشده، اما به‌طور کلی یادگیری مؤثری هم نداشته است.
3. یادگیری کند و ناکارآمد نیز از روی اعداد مشخص است: در ۴۱ دوره آموزش هنوز به کمتر از 45٪ رسیده که برای مسائل طبقه‌بندی بسیار پایین تلقی می‌شود.

1. **ضرایب اهمیت ویژگی‌ها بین دیتاست‌ها تفاوت چشمگیر دارد:**

- مدل‌های از پیش آموزش‌دیده مثلاً روی داده‌های پزشکی، اقتصادی یا صنعتی هر کدام **ویژگی‌های متفاوتی دارند**.
- بنابراین انتقال وزن‌ها از یک مسئله جدولی به مسئله‌ای دیگر **معادل با استفاده از فرضیات نادرست** در یادگیری است.

2. **Embeddingهای آموخته‌شده قابل انتقال نیستند:**

- اگر مدل بخواهد مثلاً رابطه‌ی بین "دما" و "فشار" را یاد بگیرد، وزن‌های آموخته‌شده برای مسئله‌ای با "سن" و "قد" کاربردی نخواهد داشت.

مدل ResNet1D که با وزن‌های از پیش آموزش‌دیده روی داده‌های جدولی استفاده شده، نتیجه‌ی ضعیفی نشان داده است .این امر تأییدی بر این واقعیت است که:

استفاده از مدل‌های از پیش آموزش‌دیده روی داده‌های جدولی نه تنها فایده‌ای ندارد بلکه ممکن است باعث گمراه شدن فرآیند یادگیری شود.

معرفی مدل دوم:Resnet50

در این بخش، مدل دوم مورد استفاده در فرآیند تشخیص یا طبقه‌بندی، شبکه‌ی ResNet-50 است. این مدل یکی از معماری‌های پرکاربرد و قدرتمند در حوزه‌ی یادگیری عمیق و به‌ویژه در مسائل بینایی ماشین است که برای استخراج ویژگی‌های عمیق از داده‌های پیچیده طراحی شده است. ساختار ResNet-50مبتنی بر معماری شبکه‌های باقی‌مانده (Residual Networks) بوده و شامل ۵۰ لایه یادگیرنده است.

ویژگی اصلی این معماری، استفاده از مسیرهای میان‌بری (skip connections) یا همان بلوک‌های باقی‌مانده (residual blocks) است. این مسیرهای میان‌بر به مدل اجازه می‌دهند تا گرادیان‌ها بدون ناپدید شدن یا انفجار، به لایه‌های اولیه بازگردند. به عبارت دیگر، با استفاده از این مکانیزم، شبکه قادر است لایه‌های بسیار عمیق‌تری را بدون افت عملکرد آموزش دهد، که در شبکه‌های سنتی عمیق‌تر اغلب منجر به تخریب دقت می‌شود.

ResNet-50از چندین بلوک پیچشی پیاپی به‌همراه نرمال‌سازی دسته‌ای، توابع فعال‌سازی و مسیرهای میان‌بر تشکیل شده است. در این مدل، هر بلوک شامل چندین لایه‌ی پیچشی است که با یک مسیر کوتاه به خروجی متصل می‌شوند. این ساختار امکان استخراج سلسله‌مراتبی از ویژگی‌ها را فراهم می‌کند که از ویژگی‌های سطح پایین در لایه‌های ابتدایی تا ویژگی‌های سطح بالا در لایه‌های انتهایی را شامل می‌شود.

این معماری به‌طور گسترده‌ای در مسائل طبقه‌بندی تصویر، شناسایی الگو، و حتی در حوزه‌های غیرتصویری با استفاده از نمایش‌های خاص داده مانند تبدیل سیگنال به تصویر)مانند spectrogram یا gramian matrixبه‌کار گرفته شده است. استفاده از نسخه‌ی از پیش آموزش‌دیده این مدل (pretrained)روی مجموعه داده‌هایی مانندImageNet ، امکان بهره‌برداری از دانش استخراج‌شده در مسائل عمومی و انتقال آن به مسائل خاص را با موفقیت فراهم می‌سازد.

در این پروژه، ResNet-50به‌عنوان مدل پایه مورد استفاده قرار گرفته و با اعمال تغییراتی در لایه‌های انتهایی آن، برای حل مسئله طبقه‌بندی مورد نظر تطبیق داده شده است. این تغییرات معمولاً شامل

جایگزینی لایه‌ی خروجی مدل با لایه‌هایی متناسب با تعداد کلاس‌های مسئله و گاهی اوقات آموزش مجدد لایه‌های پایانی یا حتی کل مدل می‌باشد، بسته به حجم داده و نیاز به تنظیم دقیق مدل.

ساختار مدل دوم

در این مدل، هدف استفاده از شبکه‌ی از پیش آموزش‌دیده ResNet-50 برای طبقه‌بندی داده‌هایی است که در اصل به‌صورت برداری بوده‌اند، اما برای سازگاری با ورودی مدل‌های تصویری، ابتدا به قالب تصویری تبدیل شده‌اند. از آنجایی که شبکه‌ی ResNet-50 برای پردازش تصاویر سه‌کاناله مانند تصاویر (RGB طراحی شده است، ابتدا داده‌ها با روش zero-padding به اندازه ۳۶ ویژگی رسانده شده و سپس به شکل تصویر ۶ در ۶ تغییر فرم داده شده‌اند. در ادامه، این تصاویر خاکستری به سه کانال تکرار شده‌اند تا با ساختار ورودی مدل انطباق یابند.

در گام بعد، معماری پایه‌ی ResNet-50 بدون بخش‌های نهایی لایه‌های (fully connected بارگذاری شده و به عنوان استخراج‌کننده ویژگی استفاده شده است. این بخش توانایی بالایی در استخراج الگوهای پیچیده و عمیق از داده‌های ورودی دارد، حتی اگر ورودی در اصل تصویر واقعی نباشد. این مزیت در انتقال دانش از مدل‌های آموزش‌دیده به مسائل جدید کاربرد دارد.

خروجی این مدل پایه از طریق لایه‌ای به نام Global Average Pooling کوچک‌سازی شده و به چندین لایه‌ی چگال (Dense) با تابع فعال‌سازی غیخطی، نرمال‌سازی دسته‌ای Batch Normalizationو حذف تصادفی نرون‌ها (Dropout) متصل شده است. این لایه‌ها نقش مهمی در یادگیری روابط غیخطی میان ویژگی‌ها و جلوگیری از بیش‌برازش ایفا می‌کنند.

در پایان، یک لایه‌ی چگال با تابع فعال‌سازی softmax برای پیش‌بینی احتمالات هر کلاس افزوده شده است. این لایه با توجه به تعداد کلاس‌های مسئله طراحی شده است.

برای آموزش این مدل، ابتدا تمام لایه‌های مدل پایه قفل شده‌اند تا فقط لایه‌های بالایی آموزش ببینند. این تکنیک در یادگیری انتقالی به منظور بهره‌گیری از دانش قبلی بدون تخریب آن استفاده می‌شود. بهینه‌سازی مدل با استفاده از روش آدام انجام شده و مکانیزم توقف زودهنگام (Early Stopping) نیز

برای جلوگیری از بیش‌برازش به کار گرفته شـده اسـت. نهایتاً، مدل در طول چندین دوره آموزش دیده و عملکرد آن بر اساس داده‌های اعتبارسنجی مورد ارزیابی قرار گرفته است.

نتایج مدل دوم

مقدار دقت تست به‌دست‌آمده تنها کمی بالاتر از حدس تصادفی (20٪) است. این نتیجه نشان‌دهنده آن است که مدل **ResNet50**نتوانسته به‌درستی الگوهای مؤثر را از داده‌های ورودی یاد بگیرد. دلایل احتمالی این عملکرد ضعیف عبارت‌اند از:

1. **عدم تناسب نوع داده با معماری مدل:**

مدل ResNet50 برای پردازش تصاویر طبیعی با ساختار فضایی مشخص طراحی شده است. در این پروژه، داده‌های عددی (جدولی/تبادلی) به صورت مصنوعی به تصویر تبدیل شده‌اند که فاقد ساختار مکانی (spatial structure) معتبر هستند. بنابراین، قابلیت استخراج ویژگی‌های فضایی مدل عمیق به شکل مؤثر به کار گرفته نشده است.

2. **تفاوت در اندازه ورودی:**

ResNet50برای پردازش تصاویر با ابعاد **3×224×224**طراحی شده است. در این پروژه، داده‌ها پس از padding تنها به اندازه **3×32×32**افزایش داده شده‌اند که این تغییر ابعاد ممکن است منجر به از دست رفتن اطلاعات مهم یا افزایش اعوجاج در عمق شبکه شود.

3. **عدم استفاده ازFine-Tuning :**

در این آزمایش، تمام لایه‌های مدل پایه فریز شده‌اند .این باعث شده فقط لایه‌های Fully Connectedآموزش ببینند و مدل نتواند ویژگی‌های سطح پایین و میانی را برای داده‌های جدید تطبیق دهد. این موضوع توانایی مدل برای انطباق با دامنه داده فعلی را به شدت محدود کرده است.

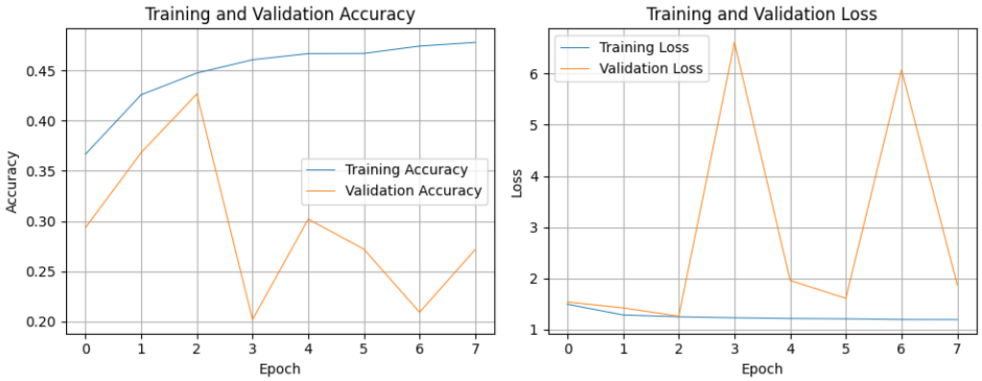
4. **فرآیند ابتدایی تبدیل داده عددی به تصویر:**

فرآیند تبدیل داده‌های عددی به تصویر تنها با padding و reshape انجام شده و فاقد هرگونه

رمزگذاری معنایی (semantic encoding) یا استخراج ویژگی‌های معنادار بوده است. در نتیجه، مدل قادر به درک تفاوت‌های کلیدی بین کلاس‌ها نبوده است.

5. **عدم تطابق ماهیت داده با نوع مدل از پیش آموزش‌دیده:**

نکته‌ی مهم‌تر اینکه دلیل اصلی عملکرد ضعیف، استفاده از مدلی است که بر روی داده‌های تصویری آموزش‌دیده است(ImageNet) ، در حالی که داده‌های ما ذاتاً جدول‌محور (tabular) هستند .بنابراین، حتی اگر مدل ResNet50 قدرتمند باشد، در غیاب ویژگی‌های بصری واقعی، توانایی آن در تفکیک کلاس‌ها محدود می‌ماند.



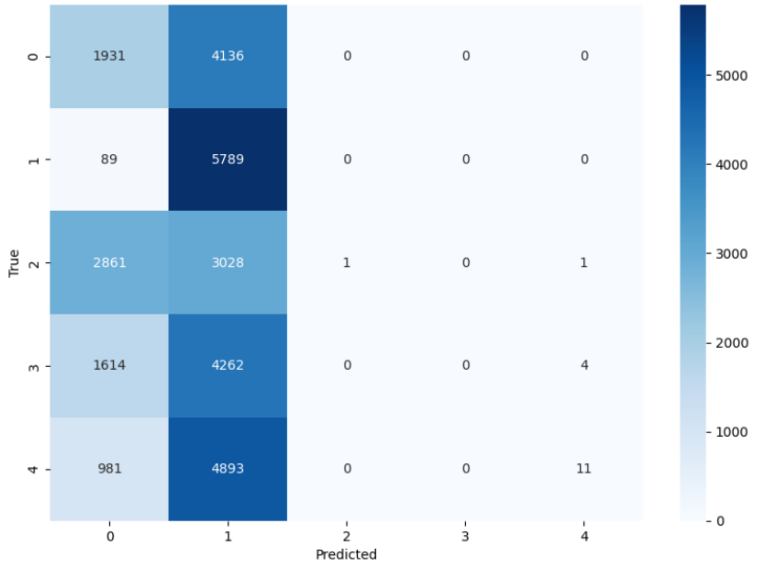
شکل 8-2 بررسی هزینه و دقت در طول آموزش مدل دوم

در نمودارهای بالا:

- **Training Accuracy**به مرور زمان افزایش یافته و به حدود 45٪ رسیده است.
- اما **Validation Accuracy**نوسان زیادی دارد و در پایان به حدود 26٪ کاهش یافته است.
- **Training Loss**کاهش یافته، اما **Validation Loss**رفتار ناپایداری دارد و افزایش قابل توجهی در انتها دارد.

تفسیر:

- این الگو نشـانه‌ای از **Overfitting**اسـت؛ مدل روی داده‌های آموزش به خوبی یاد گرفته اما در تعمیم به داده‌های جدید (اعتبارسنجی) ناتوان بوده است.
- عدم همگرایی مناسب loss اعتبارسنجی نشان می‌دهد مدل نتوانسته الگوهای کلی را یاد بگیرد.



شکل 8-3 ماتریس سردرگمی مدل دوم

ماتریس نشان می‌دهد که تقریباً اکثر نمونه‌ها صرفاً به کلاس **"AddWeight"**(برچسب 1) پیش‌بینی شده‌اند، به طوری که:

بیشتر کلاس‌ها به اشتباه به کلاس 1 نسبت داده شده‌اند.

دقت در تفکیک کلاس‌ها به‌شدت پایین است، به‌ویژه در کلاس‌های "PropellerDamage_slight" و "PressureGain_constant".

تفسیر:

مدل به دلیل **عدم توانایی در استخراج ویژگی‌های متمایز**، تمایل به پیش‌بینی غالب‌ترین کلاس در توزیع داده‌ها دارد.(bias toward majority class)

نبود ویژگی‌های بصری مشخص در تصاویر (که از داده‌های جدولی تولید شده‌اند)، موجب شده مدل نتواند تمایز بین کلاس‌ها را بیاموزد.

Support	F1-score	Recall	Precision	کلاس
6067	0.29	0.32	0.26	Normal
5878	0.41	0.98	0.26	AddWeight
5891	0.00	0.00	1.00	PropellerDamage_slight
5880	0.00	0.00	0.00	PropellerDamage_bad
5885	0.00	0.00	0.69	PressureGain_constant
29601	0.26			:Overall Accuracy

جدول 8-1 معیار های ارزیابی مدل دوم

تفسیر:

- **دقت کلی (accuracy) تنها 26٪** است.
- کلاس "AddWeight" با **دقت بالا در Recall (0.98)** شناسایی شده، ولی سایر کلاس‌ها تقریباً اصلاً شناسایی نشده‌اند.
- **F1-scoreبرای اکثر کلاس‌ها صفر یا بسیار پایین** است؛ نشان‌دهنده عدم توازن در عملکرد مدل در کلاس‌های مختلف.

معرفی مدل سوم : VGG16

در مدل سوم، از معماری معروف **VGG16**به‌عنوان یک شبکه‌ی عصبی کانولوشنی عمیق استفاده شده است. این معماری به‌طور خاص به دلیل سادگی ساختار و عمق مناسب آن در استخراج ویژگی‌های غنی از تصاویر، شناخته شده است. در این پروژه، هدف استفاده از مدل VGG16 برای طبقه‌بندی مجموعه داده‌ای است که در اصل به‌صورت برداری بوده، اما مشابه مراحل قبلی، ابتدا به شکل تصاویر کوچک تبدیل شده و سپس برای انطباق با ورودی‌های این مدل از پیش آموزش‌دیده، به تصاویر سه‌کاناله در ابعاد مناسب تبدیل شده‌اند.

مدل پایه VGG16 که از وزن‌های آموزش‌دیده روی مجموعه داده‌ی ImageNet بهره می‌برد، بدون لایه‌های بالایی (fully connected) بارگذاری شده است. این مدل دربردارنده‌ی توالی منظمی از لایه‌های کانولوشنی با کرنل‌های ۳×۳ و لایه‌های حداکثر Poolingبوده و تولنایی بالایی در تشخیص الگوهای موضعی و فضایی در تصاویر دارد.

پس از استخراج ویژگی‌ها توسط این مدل، یک بخش سفارشی شامل چندین لایه‌ی چگال به مدل افزوده شده است. این لایه‌ها همراه با تکنیک‌هایی مانند **Batch Normalization**برای پایدارسازی فرآیند آموزش، **ReLU**برای اعمال غیخطی بودن، و **Dropout**برای کاهش بیش‌برازش، طراحی شده‌اند. در انتها، یک لایه‌ی خروجی softmax برای پیش‌بینی احتمال تعلق هر نمونه به یکی از کلاس‌های مشخص شده قرار گرفته است.

برای بهره‌گیری از قدرت مدل از پیش آموزش‌دیده و جلوگیری از یادگیری مجدد کل شبکه، لایه‌های مدل پایه در ابتدا قفل شده‌اند و فقط لایه‌های افزوده‌شده آموزش دیده‌اند. از روش بهینه‌سازی **Adam** برای تنظیم وزن‌ها استفاده شده و مکانیزم **توقف زودهنگام** نیز در آموزش لحاظ شده است تا در صورت عدم بهبود عملکرد روی داده‌های اعتبارسنجی، آموزش متوقف گردد. این مدل از لحاظ دقت، پایداری و قدرت تعمیم‌پذیری، یکی از گزینه‌های مناسب برای مسائل طبقه‌بندی با داده‌های تصویری تلقی می‌شود.

ساختار مدل سوم

ساختار مدل سوم مبتنی بر معماری VGG16 طراحی شده است که ابتدا تصاویر ورودی به ابعاد 32 در 32 و با سه کانال رنگی تغییر اندازه داده می‌شوند تا با ورودی مدل هماهنگ شوند. مدل پایه VGG16 بدون لایه‌های بالایی و با وزن‌های آموزش‌دیده روی مجموعه ImageNet بارگذاری می‌شود تا بتولند ویژگی‌های سطح بالای تصاویر را به خوبی استخراج کند.

پس از عبور داده‌ها از مدل پایه، یک لایه Global Average Pooling برای کاهش ابعاد و تبدیل ویژگی‌ها به یک بردار یک‌بعدی قرار داده شده است. سپس چندین لایه چگال متوالی به مدل افزوده می‌شود که هر کدام شامل تابع فعال‌سازی ReLU ، نرمال‌سازی دسته‌ای (Batch Normalization) برای تثبیت آموزش و لایه Dropout برای کاهش بیش‌برازش هستند. اندازه این لایه‌ها به تدریج کاهش می‌یابد تا استخراج ویژگی‌ها بهینه‌تر و فشرده‌تر انجام شود.

در انتهای ساختار، یک لایه خروجی با تابع فعال‌سازی softmax وجود دارد که تعداد نورون‌های آن برابر با تعداد کلاس‌های هدف است و مسئول دسته‌بندی نمونه‌ها به کلاس‌های مختلف می‌باشد.

برای آموزش مدل، لایه‌های مدل پایه فریز شده‌اند تا در مرحله انتقال یادگیری (Transfer Learning) تنها بخش‌های افزوده‌شده به صورت بهینه آموزش ببینند و از دانش از پیش یادگرفته شده مدل پایه استفاده شود. بهینه‌ساز Adam با نرخ یادگیری مشخص به کار گرفته شده و همچنین از روش‌های توقف زودهنگام و کاهش نرخ یادگیری در صورت عدم بهبود عملکرد روی داده‌های اعتبارسنجی بهره برده شده است.

این ساختار به دلیل ترکیب استخراج ویژگی‌های عمیق توسط VGG16 و لایه‌های کاملاً متصل سفارشی، برای مسائل طبقه‌بندی تصاویر با تعداد کلاس محدود، انتخاب مناسبی به شمار می‌رود.

نتایج مدل سوم

دقت آزمون نزدیک به 63٪ نشان می‌دهد که مدل توانسته الگوهایی از داده‌ها را یاد بگیرد و تا حد قابل قبولی عملکردی بهتر از حدس تصادفی (20٪ برای 5 کلاس ارائه دهد. این نتیجه نسبت به ResNet50با دقت تنها 26٪ (به‌مراتب بهتر است و نشان از تناسب نسبی بیشتر VGG16 با داده‌های فعلی دارد.

دلایل عملکرد بهتر نسبت به ResNet50 :

ساختار VGG16 ساده‌تر است:

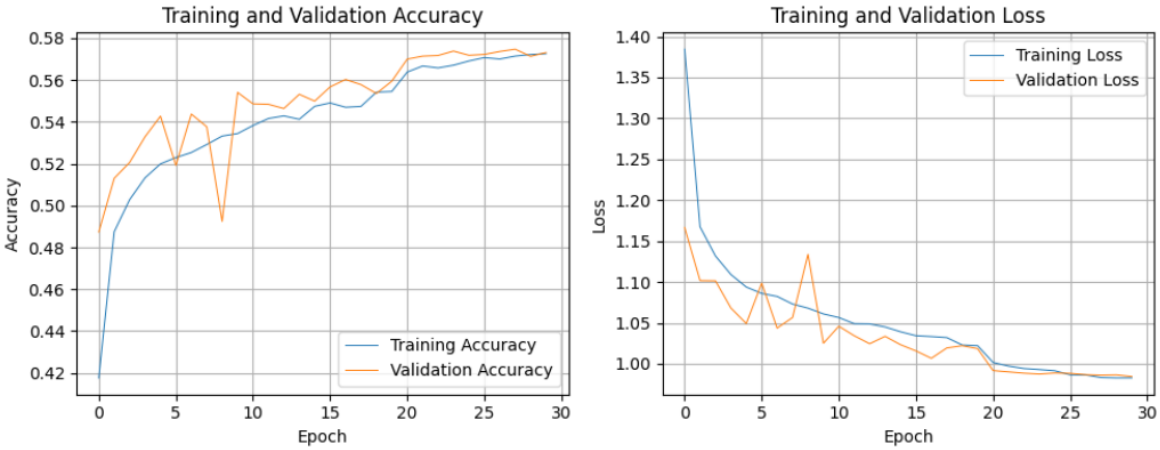
- VGG16برخلافResNet50 ، از بلوک‌های ساده Conv-ReLU-MaxPooling تشکیل شده و پیچیدگی کمتری دارد.
- این سادگی باعث شده مدل به داده‌های کم‌ساختار (مانند تصاویر تولیدشده از داده‌های جدولی) بهتر واکنش نشان دهد.

ابعاد تصویر ورودی(32×32) :

- اگرچه VGG16 نیز معمولاً برای تصاویر بزرگ‌تر آموزش دیده، اما ساختار لایه‌های آن برای تصاویر کوچک کمتر تلفات ویژگی ایجاد می‌کند نسبت به ResNet50 که در لایه‌های ابتدایی از stride بالا استفاده می‌کند.

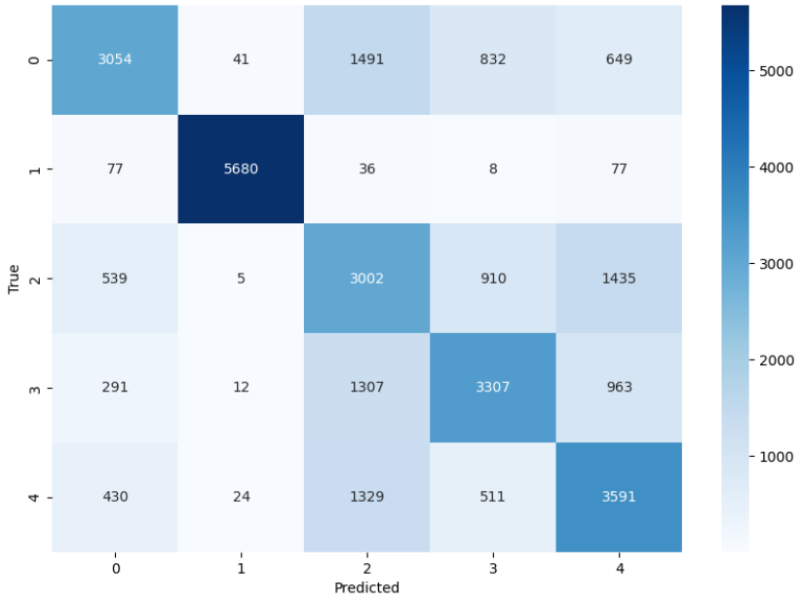
تأثیر لایه‌های Fully Connected وDropout :

- معماری بالایی شامل لایه‌های Dense با Dropout وBatchNorm ، توانسته به بهبود تعمیم مدل کمک کند.



شکل 8-4 هزینه و دقت در طول آموزش مدل سوم

- دقت آموزش از حدود 42٪ شروع شده و به حدود 57٪رسیده است.
- دقت اعتبارسنجی نیز روند صعودی نسبتاً پایدار داشته و در پایان نزدیک به دقت آموزش تثبیت شده است.
- این نزدیکی دو منحنی نشان‌دهنده‌ی عدم وجود overfitting قابل توجه در مدل است.
- خطاهای آموزش و اعتبارسنجی نیز به صورت هماهنگ کاهش یافته‌اند و در پایان هر دو به حدود 1.0رسیده‌اند.
- نبود شکاف بزرگ بین دو منحنی نشان می‌دهد که مدل از نظر توانایی تعمیم (generalization)وضعیت مناسبی دارد.



شکل 8-5 ماتریس سردرگمی مدل سوم

مدل در تشخیص کلاس‌های دارای اختلالات مشابه مانند PropellerDamage ضعف تفکیکی دارد اما در کل عملکر بهتر و قابل قبول تری نسبت به مدل های قبلی دارد.

F1-Score	Recall	Precision	کلاس
0.58	0.50	0.70	Normal
0.98	0.97	0.99	AddWeight
0.46	0.51	0.42	PropellerDamage_slight
0.58	0.56	0.59	PropellerDamage_bad
0.57	0.61	0.53	PressureGain_constant

جدول 8-2 معیار های ارزیابی مدل سوم

- کلاس AddWeightعملکرد بسیار خوبی دارد (تقریباً ایده‌آل)، در حالی که بقیه کلاس‌ها دقت و فراخوانی متوسط دارند.
- میانگین دقت و F1 برابر با 0.63تا 0.65 است، که برای داده‌های تولیدشده از ویژگی‌های جدولی (tabular) مطلوب تلقی می‌شود.

معرفی مدل چهارمMobilenet:

مدل چهارم مبتنی بر معماری موبایل‌نت (MobileNet) است که برای کاربردهای موبایل و سیستم‌های با محدودیت محاسباتی طراحی شده و تاکید زیادی بر سبک‌وزنی و سرعت اجرا دارد. این مدل از تکنیکی به نام «تفکیک‌پذیری کانال‌های پیچشی (Depthwise Separable Convolution) «بهره می‌برد که به جای اعمال یک فیلتر بزرگ روی همه کانال‌های ورودی، ابتدا هر کانال را جداگانه فیلتر می‌کند و سپس نتایج را با هم ترکیب می‌کند. این کار باعث کاهش چشمگیر تعداد پارامترها و عملیات محاسباتی می‌شود بدون آنکه افت زیادی در دقت مدل ایجاد شود.

در این روش، ابتدا تصاویر ورودی به شکل استانداردی (مثلاً 96×96 پیکسل و ۳ کانال رنگی) تغییر اندازه داده می‌شوند تا ورودی مدل را تشکیل دهند. سپس مدل MobileNet با وزن‌های آموزش‌دیده روی مجموعه ImageNet بارگذاری شده و لایه‌های بالایی آن حذف می‌شود تا بتوان لایه‌های سفارشی و خاص مسئله خود را جایگزین کرد.

پس از بخش استخراج ویژگی توسطMobileNet ، از لایه Global Average Pooling استفاده می‌شود تا ابعاد ویژگی‌ها کاهش یافته و آماده اتصال به لایه‌های کاملاً متصل شود. چند لایه Dense به همراه Batch Normalization و Dropout به مدل افزوده می‌شوند تا به کمک آنها، ویژگی‌های استخراج‌شده بهتر ترکیب و بهینه شوند و در نهایت، لایه خروجی با تابع فعال‌سازیsoftmax ، مسئول طبقه‌بندی نمونه‌ها به کلاس‌های هدف است.

در فرآیند آموزش، لایه‌های پایه مدل موبایل‌نت فریز می‌شوند تا فقط بخش‌های افزوده شده به صورت ویژه آموزش ببینند و از دانش از پیش‌آموخته بهره‌مند شوند. به این ترتیب، مدل هم سبک و سریع باقی می‌ماند و هم عملکرد مناسبی در طبقه‌بندی داده‌ها ارائه می‌دهد.

این مدل برای کاربردهایی که محدودیت منابع سخت‌افزاری و نیاز به سرعت بالا دارند، گزینه‌ای بسیار کارآمد و متداول است.

ساختار مدل چهارم

مدل چهارم مبتنی بر معماری MobileNet طراحی شده است که یک شبکه عصبی کانولوشنی سبک‌وزن و کارآمد برای دستگاه‌های دارای محدودیت منابع است. این مدل با استفاده از وزن‌های آموزش‌دیده بر روی مجموعه داده ImageNet بارگذاری می‌شود و لایه‌های بالایی آن حذف می‌گردد تا بتوان لایه‌های سفارشی را بر اساس مسئله مورد نظر اضافه کرد.

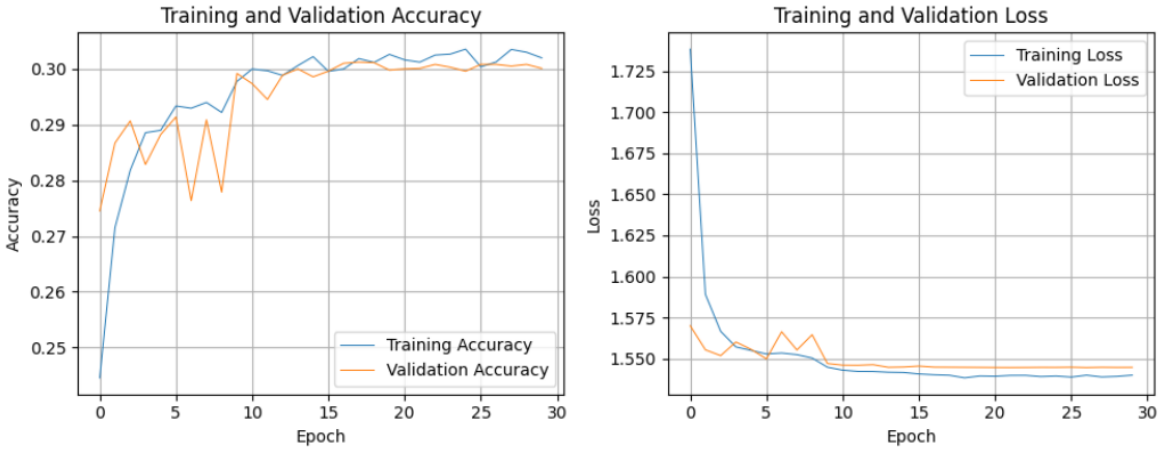
در ادامه، پس از استخراج ویژگی‌ها توسط بخش پایه مدل، از یک لایه Global Average Pooling برای کاهش ابعاد ویژگی‌ها استفاده می‌شود که به کاهش تعداد پارامترها کمک می‌کند و مدل را برای اتصال به لایه‌های کاملاً متصل آماده می‌سازد. سپس چندین لایه Dense با توابع فعال‌سازیReLU ، همراه با Batch Normalization و Dropout برای جلوگیری از بیش‌برازش به مدل افزوده می‌شود. در نهایت، لایه خروجی با تابع softmax قرار داده شده است تا مدل بتواند نمونه‌ها را در بین پنج کلاس طبقه‌بندی کند.

برای حفظ دانش از پیش‌آموخته و جلوگیری از به‌روزرسانی وزن‌های بخش پایه، تمام لایه‌های MobileNetدر مرحله آموزش فریز می‌شوند، به‌طوری که فقط لایه‌های بالاساز آموزش می‌بینند. مدل با استفاده از بهینه‌ساز Adam و تابع خطای categorical crossentropy کامپایل شده است.

در نهایت، آموزش مدل در طی چندین دوره با تنظیمات مناسب شامل Early Stopping برای جلوگیری از آموزش بیش از حد و کاهش نرخ یادگیری در صورت عدم بهبود عملکرد، انجام می‌شود. این ساختار باعث می‌شود مدل سریع، کم‌حجم و با دقت مناسب برای کاربردهای طبقه‌بندی تصویر باشد، به‌ویژه در محیط‌های محدود از نظر منابع محاسباتی.

نتایج مدل چهارم

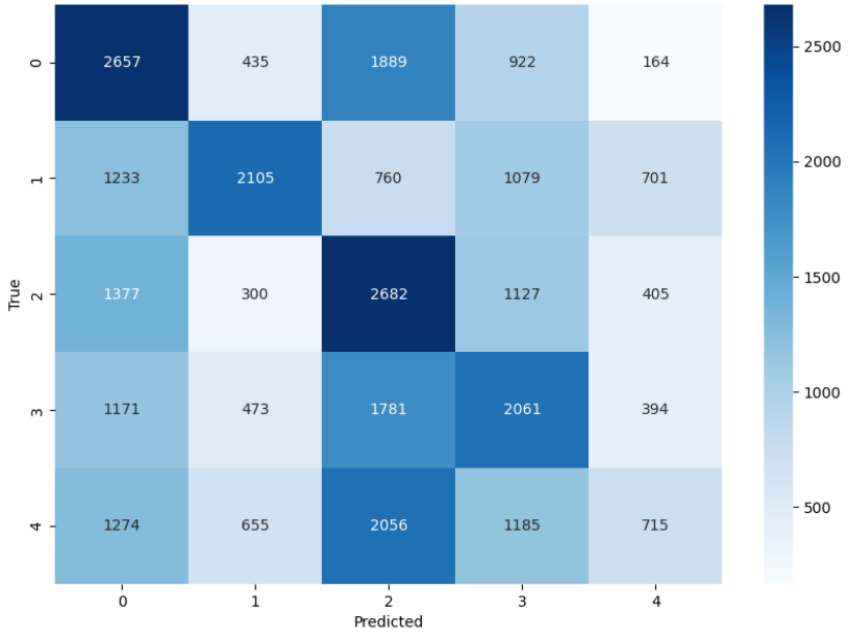
مدل MobileNet در این پروژه با دقت تست (Test Accuracy) برابر با 0.3453 عملکرد ضعیفی از خود نشان داده است. با توجه به اینکه مسأله شامل ۵ کلاس است، این دقت نزدیک به حد تصادفی (20%) نیست اما بسیار پایین‌تر از انتظار برای یک مدل مبتنی بر یادگیری عمیق مانند MobileNet است.



شکل 8-6 هزینه و دقت در طول آموزش مدل چهارم

- دقت آموزش از حدود 24٪ آغاز شده و نهایتاً به حدود 30٪رسیده است.
- دقت اعتبارسنجی نیز روندی مشابه داشته و در حوالی همان مقدار نوسان کرده است.
- منحنی‌ها بسیار صاف و یکنواخت هستند و هیچ بهبود چشمگیری در طول epoch 30 مشاهده نمی‌شود.
- مدل نتوانسته است روابط معناداری از داده‌ها یاد بگیرد.
- این احتمال وجود دارد که ویژگی‌های ورودی برای MobileNet مناسب نباشند (مثلاً داده‌های تصویری دارای ویژگی‌های تکراری یا غیرمرتبط باشند).
- هر دو منحنی از حدود 1.75 شروع شده و به حدود 1.55 کاهش یافته‌اند.

- برخلاف انتظار، افت خطا خیلی کم است و پس از epoch 5 تقریباً ثابت می‌شود.
- این نشان‌دهنده‌ی ایستایی یادگیری و ناتوانی مدل در کاهش خطا است.
- مدل MobileNet نتوانسته است به درستی یادگیری را انجام دهد.
- احتمالاً به تنظیمات هایپرپارامترها یا پیش‌پردازش داده‌ها نیاز دارد.



شکل8-7 ماتریس سردرگمی مدل چهارم

- توزیع پیش‌بینی‌ها به شدت پراکنده است.
- کلاس‌ها به‌ویژه 2، 3 و 4 (PropellerDamage) و (PressureGain دچار اختلاط شـــدید با یکدیگر هستند.
- بســـیاری از نمونه‌ها از کلاس 1 (AddWeight) و 4 (PressureGain) به عنوان کلاس‌های دیگر تشخیص داده شده‌اند.
- PropellerDamage_slightبا بیش از 1000 نمونه به عنوان کلاس‌های دیگر پیش‌بینی شده.
- کلاس PressureGain_constant تنها در 12٪ موارد درست پیش‌بینی شده که بسیار پایین است.

F1-Score	Recall	Precision	کلاس
0.39	0.44	0.34	Normal
0.43	0.36	0.53	AddWeight
0.36	0.46	0.29	PropellerDamage_slight
0.34	0.35	0.32	PropellerDamage_bad
0.17	0.12	0.30	PressureGain_constant

جدول 8-3 معیارهای ارزیابی مدل چهارم

- تنها کلاس نسبتاً قابل قبول AddWeight است(F1=0.43) ، اما باز هم از حد مطلوب فاصله دارد.
- کلاس PressureGain_constant عملکرد بسیار ضعیفی دارد.(F1=0.17)
- میانگین F1 برابر با 0.34است، که برای کاربردهای عملی قابل قبول نیست.

نتیجه گیری کلی و مقایسه مدل ها

در این پژوهش، چند مدل یادگیری عمیق با استفاده از تکنیک یادگیری انتقالی (Transfer Learning)بر روی داده‌های تصویری مورد ارزیابی قرار گرفتند. همچنین، یک مدل مبتنی بر داده‌های جدولی نیز با ResNet1D ارزیابی شد. هدف نهایی، طبقه‌بندی دقیق انواع خطاهای مرتبط با عملکرد پره‌ها و فشار در یک سیستم صنعتی بود.

خلاصه عملکرد مدل‌ها:

نام مدل	نوع داده	استفاده از Transfer Learning	دقت تست (Accuracy)	میانگین F1-score	نکات مهم
ResNet1D	جدولی (Tabular)	بله	0.42	—	یادگیری انتقالی روی داده جدولی ناکارآمد است
ResNet50	تصویری (Image)	بله	0.26	0.14	Recall سه کلاس صفر، عملکرد ضعیف
VGG16	تصویری (Image)	بله	0.63	0.63	بهترین عملکرد بین تمام مدل‌ها
MobileNet	تصویری (Image)	بله	0.35	0.34	عملکرد متوسط در تمام کلاس‌ها

جدول 8-4 مقایسه مدل ها

تحلیل عملکرد مدل‌ها:

1. VGG16
   - ✓ بهترین عملکرد در میان تمام مدل‌ها(Accuracy = 63%)
   - ✓ Precisionو Recall در تمام کلاس‌ها نسبتاً متعادل.
   - ✓ مناسب‌ترین مدل برای داده‌های تصویری این پروژه.
   - ✓ فیلترهای pretrained به‌خوبی الگوهای تصویری خطا را استخراج کرده‌اند.
2. ResNet50
   - ✓ عملکرد بسیار ضعیف با Accuracy = 26% وF1-score = 0.14
   - ✓ Recallبرای سه کلاس از جمله PropellerDamage و PressureGain صفر است.
   - ✓ نشان‌دهنده عدم توانایی مدل در تفکیک ویژگی‌های مربوط به این کلاس‌ها در تصاویر خاص این پروژه.

3. MobileNetV2
   - ✓ عملکرد ضعیف تا متوسط(Accuracy = 35%)
   - ✓ Precisionنسبتاً بهتر، اما Recall برای بعضی کلاس‌ها) مثل PressureGain_constant) پایین.
   - ✓ معماری سبک ممکن است نتوانسته باشد اطلاعات پیچیده‌ی تصویری را به‌خوبی استخراج کند.
4. ResNet1D
   - ✓ عملکرد نسبتاً قابل قبول(Accuracy = 42%)
   - ✓ اما استفاده از یادگیری انتقالی برای داده‌های جدولی بی‌معنی است.

چرا Transfer Learning برای داده‌های جدولی بی‌معناست؟

داده‌های جدولی ساختار مکانی یا زمانی واضحی ندارند
در حالی که مدل‌هایی مانند ResNet یا VGG مخصوص داده‌های تصویری با الگوهای مکانی‌اند.

ویژگی‌ها در داده‌های جدولی معنادار هستند اما مستقل از ساختار شبکه کانولوشنی‌اند
مدل‌های pretrained نمی‌توانند اهمیت فیزیکی و معنایی ویژگی‌ها (مثل فشار، سرعت، یا دما) را درک کنند.

شبکه‌های pretrained دارای وزنی از پیش‌تنظیم‌شده‌اند
این وزن‌ها برای داده‌های خاص) مانند (ImageNet بهینه شده‌اند و در داده‌های جدولی ممکن است منجر به عملکرد گمراه‌کننده شوند.

نتیجه نهایی و پیشنهادات

- برای داده‌های تصویری:
  - ✓ از میان مدل‌ها، VGG16بهترین انتخاب برای طبقه‌بندی دقیق و متوازن خطاهاست.
  - ✓ ResNet50و MobileNet به‌دلیل ساختار خاص داده یا تنظیمات مدل، عملکرد مطلوبی نداشتند.
- **برای داده‌های جدولی:**
  - ✓ پیشنهاد می‌شود به‌جای Transfer Learning از مدل‌هایی مانند:
    - ❖ MLPهای بهینه‌شده
    - ❖ XGBoost / LightGBM
    - ❖ TabNetیا FT-Transformer

    استفاده شود که برای ساختار غیرتصویری طراحی شده‌اند.

لازم به ذکر است که مدل های بالا نیز نباید با روش transfer learning پیاده سازی شوند.

فصل نهم:

عیب یابی مبتنی بر RL

معرفی روش

با پیچیده‌تر شدن سیستم‌های کنترلی و مکانیکی در حوزه‌هایی همچون هوافضا، رباتیک، سامانه‌های دریایی و صنعتی، نیاز به روش‌هایی هوشمند و تطبیقی برای شناسایی و مقابله با عیب‌ها بیش از پیش احساس می‌شود. یکی از رویکردهای نوین در این زمینه، بهره‌گیری از الگوریتم‌های یادگیری تقویتی (RL)است. یادگیری تقویتی، که از شاخه‌های اصلی یادگیری ماشین به شمار می‌رود، به عامل (agent) این امکان را می‌دهد تا از طریق تعامل با محیط و دریافت پاداش، راهبردی بهینه برای انجام وظیفه‌ای خاص بیاموزد.

در مسئله‌ی عیب‌یابی، هدف این است که عامل یاد بگیرد چگونه با مشاهده داده‌های محیط یا خروجی سیستم، وجود یک یا چند نوع عیب را شناسایی کند و در صورت امکان، تصمیمی بهینه برای جبران یا کنترل اثر آن عیب اتخاذ نماید. مزیت اصلی یادگیری تقویتی نسبت به روش‌های سنتی در این است که نیازی به مدل دقیق سیستم یا عیب ندارد و می‌تواند در محیط‌های پیچیده و نامعین نیز عملکرد قابل قبولی ارائه دهد.

در این پروژه، تلاش شد تا با پیاده‌سازی الگوریتم **Proximal Policy Optimization (PPO)**یکی از الگوریتم‌های پیشرفته در خانواده سیاست‌محور (policy-based) یادگیری تقویتی، مدلی طراحی شود که بتواند بر اساس داده‌های دریافتی از یک محیط شبیه‌سازی‌شده، اقدام به تشخیص یا تطبیق در برابر خطاهای احتمالی کند. با وجود چالش‌هایی در فرایند آموزش، این رویکرد توانست دیدگاه مفیدی درباره‌ی ظرفیت‌های RL در حوزه‌ی عیب‌یابی فراهم سازد.

شکل 9-1 عیب یابی AUV با استفاده از RL

آماده‌سازی داده‌ها

در این پروژه از پنج دسته داده مربوط به شرایط مختلف عملکردی و خطای یک سامانه بهره‌برداری شده است. این شرایط شامل حالت نرمال(Normal) ، افزایش وزن(AddWeight) ، آسیب خفیف به پروانه (PropellerDamage_slight)، آسیب شدید به پروانه (PropellerDamage_bad) و افزایش فشار ثابت (PressureGain_constant) می‌باشند. برای هر کلاس، تعداد ۵۰ فایل ثبت داده به‌صورت مجزا در نظر گرفته شد و داده‌ها پس از تجمیع، به‌صورت یک آرایه پیوسته جهت آموزش مدل مورد استفاده قرار گرفتند.

هر نمونه داده دارای ۱۷ ویژگی (feature) عددی بود که از حسگرهای مختلف سیستم جمع‌آوری شده است. برخلاف برخی رویکردهای رایج در پروژه‌های پیشین، در این مطالعه از اعمال تبدیل فوریه (FFT)بر روی داده‌ها خودداری شد. دلیل این تصمیم، دو نکته کلیدی بود:

1. حفظ حجم کامل داده‌ها :تبدیل فوریه معمولاً باعث کاهش بُعد زمانی داده‌ها و از بین رفتن جزئیاتی در فرکانس پایین یا بالا می‌شود. از آنجا که قصد داشتیم تمامی ویژگی‌های موجود در داده حفظ شوند و از کاهش حجم اطلاعات جلوگیری شود، از استفاده از FFT صرف‌نظر شد.
2. تجربه موفق پیشین با CNN بدونFFT : در پروژه‌های پیشین نیز مشاهده شد که شبکه‌های عصبی کانولوشنی (CNN) حتی بدون استفاده از تبدیل فوریه عملکرد مناسبی در استخراج ویژگی از داده‌های خام داشته‌اند. بنابراین، فرض بر این بود که داده‌های خام نیز برای آموزش مدلRL ، به‌ویژه در مرحله استخراج ویژگی، کافی هستند.

برای افزایش تنوع داده‌ها و شبیه‌سازی شرایط متغیر در داده‌های حسگر، به داده‌ها نویز گوسی با واریانس کنترل‌شده اضافه شد. این کار باعث افزایش قابلیت تعمیم (generalization) مدل در برابر نویز و داده‌های دنیای واقعی می‌شود.

همچنین داده‌ها قبل از ورود به مدل با استفاده از استانداردسازی (StandardScaler) نرمال شدند تا میانگین داده‌ها به صفر و انحراف معیار به یک برسد. این فرآیند موجب تسهیل در یادگیری مدل‌های مبتنی بر گرادیان مانند PPO می‌شود.

در پایان، داده‌ها جهت استفاده در مدل به شکلی که هر نمونه فقط شامل ۱۷ ویژگی عددی باشد، بازشکل‌دهی (reshape) شدند و داده‌های آموزش و آزمون به‌صورت جداگانه آماده‌سازی گردیدند.

طراحی محیط تقویتی AUV برای تشخیص عیب

ایجاد محیط و منطق پاداش‌دهی

در این پروژه، برای پیاده‌سازی سیستم تشخیص عیب مبتنی بر یادگیری تقویتی، یک محیط سفارشی طراحی شد که مطابق با قالب استاندارد کتابخانه Gymپیاده‌سازی شده است. این محیط داده‌های از پیش آماده‌شده‌ی مربوط به پنج کلاس مختلف از شرایط عملکردی یک وسیله‌ی زیرسطحی خودگردان (AUV)را دریافت می‌کند و در هر مرحله، یک نمونه تصادفی را به عامل (Agent) ارائه می‌دهد. عامل موظف است بر اساس مشاهده‌ی داده، کلاس صحیح (یعنی نوع عیب یا شرایط نرمال) را تشخیص دهد و بر اساس درستی یا نادرستی این تصمیم، پاداش دریافت کند.

شیوه‌ی پاداش‌دهی پویا(Dynamic Rewarding)

طراحی سیستم پاداش‌دهی یکی از مهم‌ترین بخش‌های یک محیط یادگیری تقویتی است و به‌شدت بر نرخ یادگیری و عملکرد عامل تأثیر می‌گذارد. در جریان این پروژه، چندین روش مختلف برای طراحی پاداش آزموده شد. از جمله:

پاداش ثابت:(Fixed reward)

در این روش، اگر پیش‌بینی عامل صحیح بود، پاداش ثابتی مثل 1+ و در صورت پیش‌بینی اشتباه، جریمه‌ای مثل 1- داده می‌شد.

مشکل :این روش به شدت باعث سوگیری مدل به سمت کلاس‌هایی با فراوانی بیشتر می‌شد و یادگیری برای کلاس‌های کم‌تعداد دشوار بود.

پاداش بر اساس توزیع کلاس‌ها:(Class frequency-based reward)

در این روش وزن هر کلاس به صورت معکوس با فراوانی آن تنظیم می‌شد. اما این روش نیز در عمل باعث افت عملکرد در طبقه‌بندی کلی شد، چراکه دقت مدل برای کلاس‌های غالب کاهش پیدا می‌کرد.

انتخاب نهایی: پاداش‌دهی پویا با وزن‌دهی تطبیقی به کلاس‌ها

در نهایت، پاداش‌دهی پویا بر اساس دقت فعلی مدل در هر کلاس انتخاب شد. در این روش، در هر مرحله، دقت جداگانه‌ای برای هر کلاس به‌صورت نسبی محاسبه می‌شود (یعنی نسبت پیش‌بینی‌های صحیح به کل نمونه‌های دیده‌شده از آن کلاس). سپس، وزن پاداش هر کلاس به صورت معکوس با دقت فعلی آن تعیین می‌شود.

یعنی:

اگر مدل تاکنون در طبقه‌بندی یک کلاس عملکرد ضعیفی داشته باشد (دقت پایین)، وزن آن کلاس بیشتر خواهد بود.

اگر مدل برای یک کلاس خاص عملکرد خوبی داشته و دقت بالایی کسب کرده باشد، وزن آن کمتر در نظر گرفته می‌شود.

این وزن‌ها در هر مرحله به صورت پویا بازمحاسبه شده و نرمال‌سازی می‌شوند.

فرمول نهایی پاداش به این شکل است:

$$\text{Reward} = \begin{cases} +\text{Weight}[y_{\text{true}}], & \text{if the prediction is correct} \\ -0.5 \times \text{Weight}[y_{\text{true}}], & \text{otherwise} \end{cases}$$

که در آن:

Weight[y_true]وزن مربوط به کلاس حقیقی است.

ضریب جریمه (penalty factor) به صورت ثابت و برابر با 0.5 در نظر گرفته شده است.

مزایای این روش پاداش‌دهی:

مقابله با عدم توازن داده‌ها :کلاس‌هایی که کمتر یا دشوارتر تشخیص داده می‌شوند، پاداش بیشتری به عامل می‌دهند و باعث هدایت توجه مدل به سمت آن‌ها می‌شوند.

سازگاری پویا با عملکرد عامل :بر خلاف روش‌های ایستا، این پاداش در طول آموزش به صورت خودکار با تغییر دقت عامل در هر کلاس به‌روزرسانی می‌شود.

کاهش سوگیری :از آنجا که وزن‌ها در طول آموزش نرمال می‌شوند، از تمرکز بیش از حد بر روی کلاس‌های خاص جلوگیری می‌شود.

پایداری در آموزش :این روش منجر به پایداری و همگرایی بهتر در طول اپیزودهای آموزشی شد و نوسان‌های پاداش کاهش یافت.

مدل اول : DQN + Resnet Feature Extractor

1. استفاده از DQN برای دسته‌بندی

معمولاً DQN برای یادگیری سیاست بهینه در مسائل تصمیم‌گیری ترتیبی استفاده می‌شود، اما در این آزمایش، از آن برای **مسئله طبقه‌بندی** استفاده شده است. ایده این است که عاملRL ، بر اساس مشاهده فعلی (شامل ۳ کانال از ویژگی‌های ۱۷تایی)، یک عمل انتخاب کند که این عمل در واقع **برچسب کلاس خطا**)مثل "Normal" یا ("Propeller Damage" باشد. اگر این عمل با برچسب واقعی تطابق داشته باشد، به عامل پاداش مثبت داده می‌شود؛ در غیر این صورت پاداش منفی دریافت می‌کند.

2. استخراج ویژگی با ResNet سفارشی

برای استخراج ویژگی‌های باکیفیت از داده‌های ورودی، از یک شبکه عصبی مبتنی بر معماری **Residual**استفاده شده است. شبکه ResNet مزیت مهمی دارد:

با اضافه کردن مسیر میانبر(shortcut) ، مشکل ناپدید شدن گرادیان‌ها در شبکه‌های عمیق را کاهش می‌دهد و یادگیری پایدارتر و دقیق‌تری به ارمغان می‌آورد.

معماری پیشنهادی شامل چند بلوک residual است که هرکدام از چند لایه خطی، نرمال‌سازی (LayerNorm)و تابع فعال‌سازی ReLU تشکیل شده‌اند. این ویژگی‌ها پس از عبور از چند مرحله استخراج، به یک بردار ویژگی ۱۷بعدی نهایی تبدیل می‌شوند که به عنوان ورودی به شبکه DQN داده می‌شود.

3. ساختار آموزش و پارامترهای یادگیری

برخی از مهم‌ترین تنظیمات آموزشی در این آزمایش عبارتند از:

- **نرخ یادگیری پایین (0.00001)** برای جلوگیری از نوسانات شدید در آموزش
- **اندازه دسته بزرگ (128)** برای بهبود پایداری گرادیان‌ها
- **حافظه تجربی بزرگ (100000)** برای حفظ تجربیات متنوع‌تر عامل
- **تکرار به‌روزرسانی هدف هر 500 مرحله** برای پایداری یادگیری-Q تابع
- **اکتشاف تدریجی** :عامل در ابتدا با احتمال ۱ به‌صورت کاملاً تصادفی عمل می‌کند و به‌تدریج این احتمال به ۰.۰۱ کاهش می‌یابد، تا عامل بتواند بین اکتشاف و بهره‌برداری تعادل برقرار کند

4. ارزیابی مدل حین آموزش با Callback سفارشی

برای پایش عملکرد مدل در طول آموزش، از یک **Callbackسفارشی** استفاده شده است. این callbackدر هر اپیزود، پاداش‌ها و دقت‌های طبقه‌بندی را ثبت می‌کند، و هر چند هزار گام، نمودارهایی از روند یادگیری رسم می‌کند. همچنین اگر میانگین پاداش در ۱۰۰ اپیزود اخیر بهتر از قبل باشد، مدل ذخیره می‌شود.

علاوه بر آن، **ماتریس درهم‌ریختگی (Confusion Matrix)** بر روی داده‌های آموزشی ترسیم می‌شود تا مشخص شود مدل در شناسایی کدام کلاس‌ها عملکرد خوبی دارد و در کدام‌ها ضعیف است.

5. نتایج و ذخیره مدل

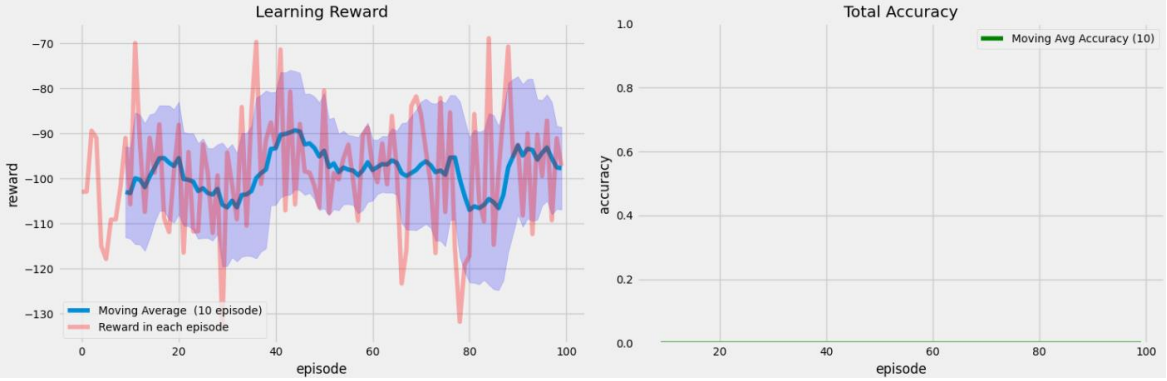
پس از پایان 50000 گام آموزشی، مدل نهایی ذخیره شده تا در آینده برای تست روی داده‌های جدید یا استقرار در سیستم‌های واقعی AUV مورد استفاده قرار گیرد. این مدل قادر است تنها با دریافت داده‌های لحظه‌ای از محیط، نوع خطای موجود در سیستم را تشخیص دهد.

نتایج مدل اول

در مدل اول، که با استفاده از الگوریتم DQN پیاده‌سازی شده بود، دقت نهایی به صورت زیر به‌دست آمد:

- دقت یادگیری25.14%
- دقت آزمون25.28%

این مقادیر نشان می‌دهند که مدل عملکرد ضعیفی در یادگیری و تعمیم داشته است .با توجه به وجود پنج کلاس در داده، این سطح از دقت نزدیک به حد تصادفی است و بیانگر آن است که مدل نتوانسته الگوهای معناداری از داده‌ها استخراج کند .همچنین نزدیکی دقت آموزش و آزمون نشان می‌دهد که مدل حتی روی داده‌های آموزشی نیز به درستی یاد نگرفته است. این نتایج نیاز به بازنگری در معماری شبکه، تنظیمات تقویتی، یا روش‌های پیش‌پردازش را نشان می‌دهد.

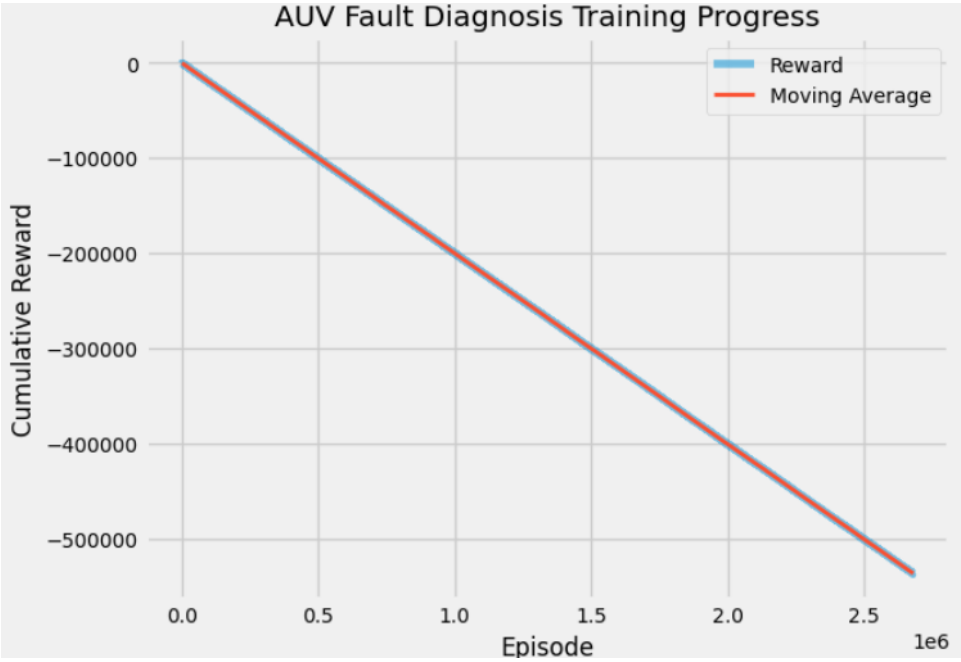


شکل 9-2 نمودار دقت و پاداش در طول آموزش مدل اول

در مدل اول که با استفاده از الگوریتم DQN پیاده‌سازی شده بود، دو نمودار مهم برای ارزیابی عملکرد مدل ترسیم شد :نمودار پاداش اپیزودها و نمودار دقت در طبقه‌بندی .بررسی این نمودارها نشان داد که عملکرد مدل در طول زمان رضایت‌بخش نبوده و پیشرفت محسوسی مشاهده نمی‌شود.

نمودار پاداش اپیزودها

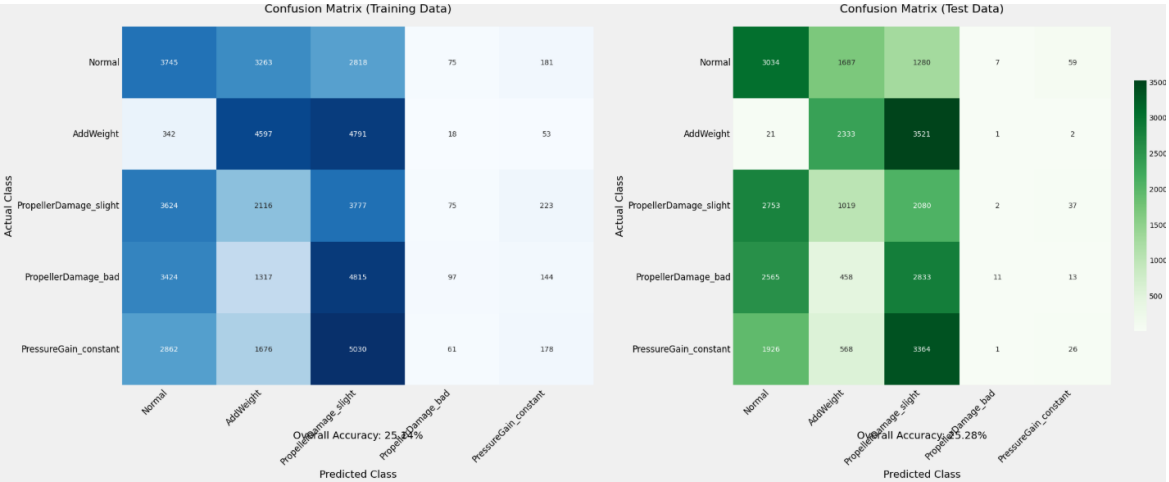
نمودار تغییرات پاداش اپیزودها (Episode Rewards) نشان می‌دهد که مقدار پاداش در طی آموزش نوسانات زیادی دارد و میانگین متحرک آن نیز افزایش قابل توجهی نداشته است .همچنین انحراف معیار پاداش در بازه‌های زمانی مختلف نسبتاً بالا باقی مانده است. این موضوع نشان می‌دهد که مدل نتوانسته یک سیاست پایدار و مؤثر برای افزایش پاداش در محیط یادگیری پیدا کند.



شکل 9-3 نمودار پاداش در طول آموزش مدل اول

همانطور که مشاهده میکنید سیستم رو به ناپایداری رفته است.

با برخی از تعاریف پاداش در محیط میتوان میزان پاداش را به سمت مثبت سوق داد اما از طرفی تنوع پیشبینی ها کم میشود و تاثیر منفی تری میگذارد.



شکل 9-4 ماتریس سردرگمی مدل اول

ماتریس سردرگمی مدل اول نشان می‌دهد که عملکرد مدل در تفکیک کلاس‌ها بسیار ضعیف بوده است. الگوی خاصی در پیش‌بینی‌ها دیده نمی‌شود و بیشتر مقادیر ماتریس در ستون‌ها و ردیف‌های نامرتبط پراکنده هستند. این موضوع بیانگر آن است که مدل هیچگونه درک معناداری از ساختار داده یا تفاوت بین کلاس‌ها پیدا نکرده است.

در واقع، توزیع پیش‌بینی‌ها به گونه‌ای است که گویی مدل هیچ آموزشی ندیده یا صرفاً به صورت تصادفی عمل کرده است .این مسئله نشان می‌دهد که یا داده‌ها برای مدل مناسب آماده‌سازی نشده‌اند، یا معماری و تنظیمات مدل توانایی یادگیری این مسئله را نداشته است. در نتیجه، خروجی مدل نه تنها قابل اتکا نیست، بلکه نشانه‌ای از نیاز جدی به بازنگری در مراحل طراحی و آموزش آن دارد.

مدل دوم : Proximal Policy Optimization (PPO)

در مدل دوم، از الگوریتم **Proximal Policy Optimization (PPO)**برای آموزش عامل تقویتی استفاده شد. این الگوریتم از خانواده روش‌های مبتنی بر سیاست (Policy-based) و روش‌های-Actor Criticاست که هدف آن بهینه‌سازی سیاست عامل با محدودسازی میزان تغییرات در هر مرحله به‌روزرسانی می‌باشد.

1. ساختار شبکه عصبی

برای طراحی شبکه، از معماری **Actor-Critic**با لایه‌های اشتراکی استفاده شد. شبکه شامل دو بخش اصلی است:

بخش مشترک(Shared layers) : شامل دو لایه Fully Connected با اندازه ۱۲۸ نورون و تابع فعال‌سازی ReLU است.

بخش بازیگر(Actor head) : یک لایه خطی که logits مربوط به توزیع اعمال ممکن را تولید می‌کند.

بخش منتقد :(Critic head) یک لایه خطی که مقدار ارزش (value) حالت فعلی را تخمین می‌زند.

2. پیاده‌سازی عاملPPO

یک کلاس تحت عنوان PPOAgentتعریف شد که شامل موارد زیر است:

مقداردهی اولیه به شبکهActor-Critic

تعیین بهینه‌ساز Adam با نرخ یادگیری اولیه lr = 3e-4

تعریف پارامترهای مربوط به تخفیف پاداش‌ها (gamma = 0.99) و پارامتر برش clip_eps = 0.2

متد get_actionبرای انتخاب عمل با استفاده از توزیع گسسته Categorical بر اساس logits خروجی شبکه

3. جمع‌آوری داده از محیط

در هر گام از تعامل عامل با محیط:

- حالت فعلی به شبکه داده می‌شود.

- عمل پیشنهادی با استفاده از نمونه‌گیری از توزیع احتمال محاسبه می‌شود.
- اطلاعات مرتبط از جمله: حالت، عمل، لاگ احتمال، مقدار ارزش(critic) ، پاداش، و وضعیت پایان اپیزود ذخیره می‌شوند.

4. فرآیند بروزرسانی سیاست

پس از رسیدن به تعداد مشخصی از گام‌ها (مثلاً 2000 گام)، تابع update_policyبرای بروزرسانی شبکه اجرا می‌شود. مراحل این بروزرسانی به شرح زیر است:

- محاسبه پاداش‌های تنزیل‌شده با استفاده از پارامتر gamma.
- محاسبه مزیت (Advantage) از تفاضل بین پاداش‌ها و مقدارهایcritic .
- محاسبه نسبت احتمال جدید به احتمال قبلی برای کنترل میزان تغییر سیاست.
- تعریف تابع هزینه شامل سه بخش:
- زیان بازیگر (actor loss) بر اساس مقدار کمینه شده نسبت‌ها
- زیان منتقد (critic loss) بر اساس خطای مقدار ارزش
- زیان انتروپی برای افزایش تنوع در انتخاب اعمال
- اجرای الگوریتم پس‌انتشار و به‌روزرسانی وزن‌های شبکه با استفاده ازoptimizer

5. فرآیند آموزش

تابع train_ppoوظیفه اجرای حلقه آموزش را بر عهده دارد:

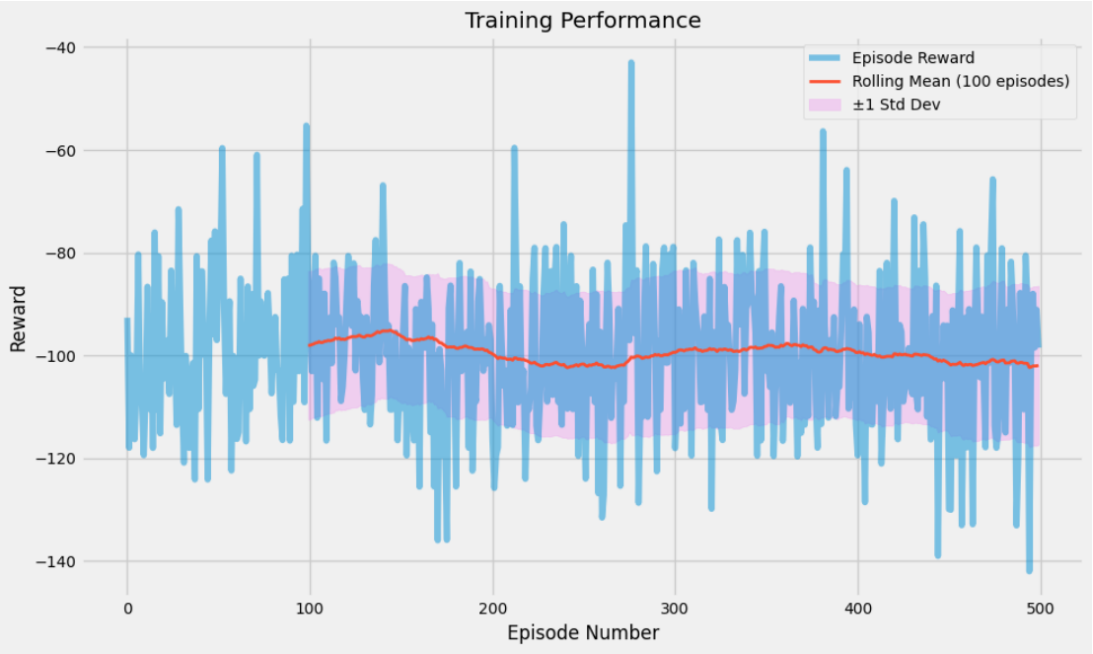
- به ازای هر اپیزود، عامل از وضعیت اولیه شروع می‌کند.
- تا پایان اپیزود، عامل با محیط تعامل می‌کند و داده‌ها را در حافظه ذخیره می‌کند.
- در بازه‌های زمانی مشخص (طبق update_interval)، شبکه به‌روزرسانی می‌شود و حافظه پاک‌سازی می‌گردد.
- در پایان هر اپیزود، مجموع پاداش دریافتی ثبت می‌شود.
- در پایان آموزش، لیستی از مجموع پاداش‌های دریافتی در هر اپیزود بازگردانده می‌شود که می‌توان از آن برای تحلیل عملکرد عامل استفاده کرد.

نتایج مدل دوم

در مدل دوم که با استفاده از الگوریتم PPO پیاده‌سازی شده بود، پس از پایان فرآیند آموزش، دقت مدل بر روی مجموعه‌های آموزش و تست محاسبه گردید:

- دقت یادگیری22.35%
- دقت آزمون22.37%

این مقادیر نشان می‌دهند که مدل عملکرد ضعیفی در طبقه‌بندی داده‌ها داشته و در تشخیص الگوهای مؤثر ناتوان بوده است .با توجه به اینکه داده‌ها شامل پنج کلاس مختلف هستند، این سطح از دقت تقریباً معادل با انتخاب تصادفی (random guessing) است و بیانگر آن است که مدل نتوانسته اطلاعات مفیدی از داده‌ها استخراج کند.



شکل 9-5 بررسی مدل دوم در طول آموزش

در این بخش، برای ارزیابی کیفیت آموزش عامل تقویتی مبتنی بر الگوریتمPPO ، مجموع پاداش‌های دریافتی در هر اپیزود رسم و تحلیل شده است.

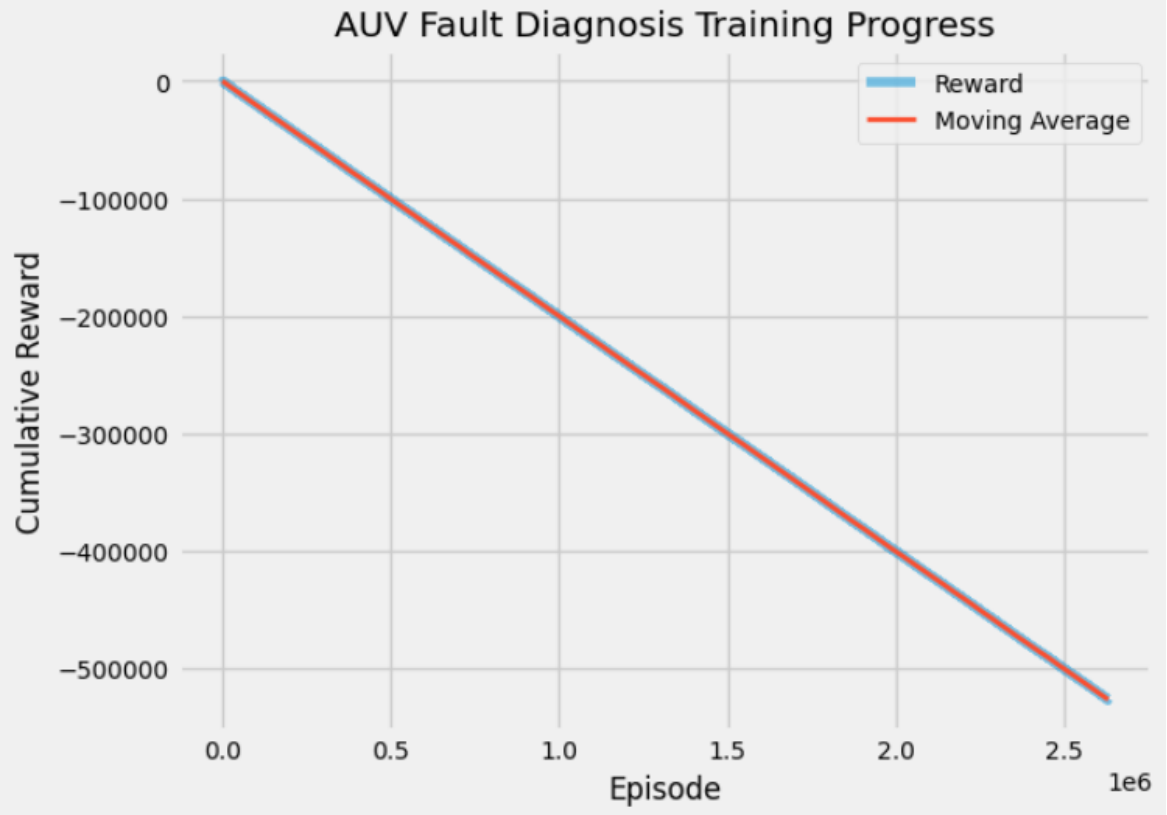
در نمودار ترسیم‌شده:

- خط آبی کم‌رنگ بیانگر مقدار پاداش دریافتی در هر اپیزود به صورت خام است.
- خط پررنگ بنفش میانگین متحرک پاداش‌ها را با پنجره‌ای به طول ۱۰۰ اپیزود نشان می‌دهد که روند کلی عملکرد عامل را بهتر آشکار می‌کند.
- ناحیه سایه‌دار بنفش نیز نشان‌دهنده‌ی انحراف معیار ± 1 حول میانگین متحرک است که میزان نوسان و عدم قطعیت عملکرد را در طول آموزش مشخص می‌سازد.

با بررسی این نمودار، مشاهده می‌شود که در طول اپیزودهای مختلف، روند مشخصی از بهبود عملکرد عامل قابل مشاهده نیست .به عبارت دیگر:

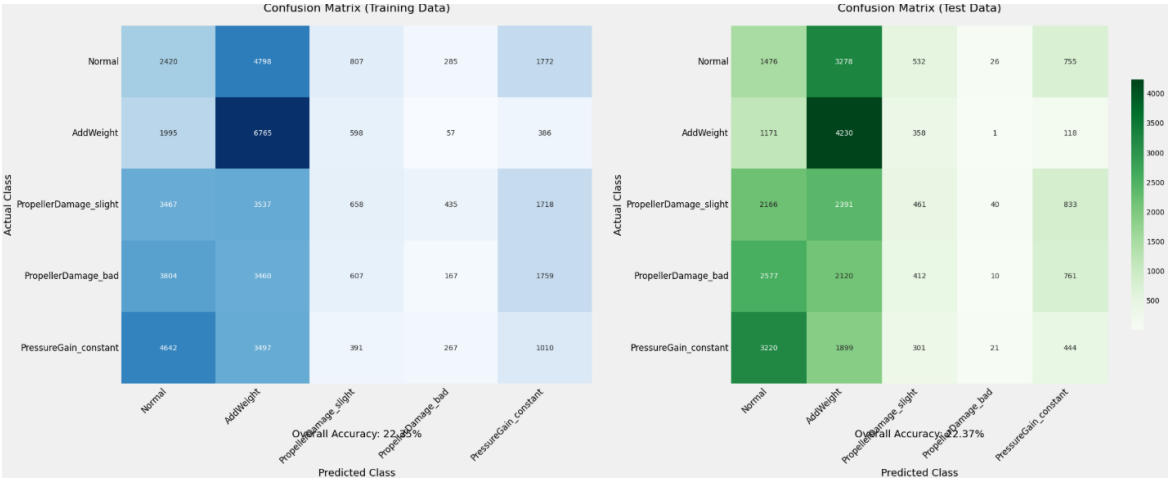
- پاداش‌ها دارای نوسانات زیاد و پراکندگی بالا هستند.
- میانگین متحرک پاداش‌ها در محدوده نسبتاً ثابتی باقی مانده و افزایش چشمگیری در آن مشاهده نمی‌شود.
- ناحیه انحراف معیار نیز نشان‌دهنده عدم پایداری در یادگیری سیاست عامل است.

این وضعیت نشان می‌دهد که مدل در شرایط فعلی نتوانسته سیاست مناسبی برای بهینه‌سازی پاداش در محیط تقویتی یاد بگیرد. ممکن است دلایل مختلفی مانند انتخاب نامناسب پارامترها (نرخ یادگیری، پنجره بروزرسانی، تعداد اپیزودها)، ساختار شبکه یا ماهیت خود محیط، بر این موضوع تأثیر گذاشته باشد.



شکل 9-6 بررسی پاداش در طول آموزش مدل دوم

همانطور که مشاهده می کنید هیچ پیشرفتی در میزان پاداش مشاهده نمی شود و حتی سیستم رو به ناپایداری رفته است.



شکل 9-7 ماتریس سردرگمی مدل دوم

در مدل دوم که با استفاده از الگوریتم Proximal Policy Optimization (PPO) پیاده‌سازی شده است، برای ارزیابی عملکرد طبقه‌بندی مدل، از ماتریس سردرگمی (Confusion Matrix) استفاده شد. بررسی نتایج این ماتریس نشان داد که مدل در تشخیص صحیح کلاس‌های مختلف عملکرد بسیار ضعیفی داشته است.

به طور مشخص:

- بیشتر پیش‌بینی‌ها در یک یا چند کلاس خاص متمرکز شده‌اند.
- برخی از کلاس‌ها اصلاً به‌درستی تشخیص داده نشده‌اند و هیچ نمونه‌ای از آن‌ها به درستی طبقه‌بندی نشده است.
- توزیع اشتباهات به صورت تصادفی به نظر می‌رسد، به‌طوری که مدل قادر به یافتن الگوی مشخص یا قاعده‌ای از داده‌ها نبوده است.

این وضعیت نشان‌دهنده‌ی این است که مدل نتوانسته است ساختار یا الگوی مشخصی در داده‌ها یاد بگیرد و تصمیم‌گیری آن تقریباً بدون اطلاعات واقعی از ویژگی‌های داده انجام شده است. در نتیجه، می‌توان گفت که یادگیری مؤثری در مدل دوم رخ نداده و عملکرد آن تقریباً به سطح حدس تصادفی نزدیک است.

- دلایل احتمالی این ضعف می‌تواند شامل موارد زیر باشد:
- طراحی نامناسب ساختار شبکه یا پارامترهای آن
- نبود بازخورد طبقه‌بندی مشخص در فرآیند یادگیری تقویتی(RL)
- ناکافی بودن اطلاعات بازخورد (reward) برای تفکیک بین کلاس‌ها
- عدم انطباق معماری PPO با وظیفه‌ی طبقه‌بندی

نتایج و مقایسه مدل ها

هر دو مدل پیاده‌سازی‌شده مبتنی بر یادگیری تقویتی DQN) و (PPO در مسئله تشخیص عیب عملکرد ضعیفی داشتند. دقت نهایی مدل DQN در داده‌های آموزش و آزمون حدود **25٪** و برای مدل PPO حدود **22٪** بود. این مقادیر به‌وضوح نزدیک به حد تصادفی (Random Guessing) هستند و نشان‌دهنده عدم موفقیت مدل‌ها در یادگیری الگوهای مؤثر از داده‌ها می‌باشند.

ماتریس‌های سردرگمی نیز در هر دو مدل بیانگر این نکته بودند که هیچ‌گونه الگوی مشخص یا تسلطی در تشخیص صحیح کلاس‌ها وجود ندارد. پیش‌بینی‌ها در کلاس‌های نادرست پراکنده شده‌اند و در بیشتر موارد مدل‌ها به صورت تصادفی عمل کرده‌اند.

با توجه به این شواهد، می‌توان نتیجه گرفت که استفاده از الگوریتم‌های **یادگیری تقویتی** برای این مسئله خاص – یعنی **تشخیص عیب از روی داده‌های زمان-فرکانس استخراج‌شده از سیگنال‌های AUV – موفقیت‌آمیز نبوده است**.

یکی از دلایل اصلی این ناکارآمدی می‌تواند **شباهت زیاد بین انواع مختلف عیب‌ها** در فضای ویژگی استخراج‌شده باشد. در صورتی که ویژگی‌های ورودی نتوانند تفکیک مناسبی بین کلاس‌های عیب ایجاد کنند، یادگیری تقویتی نیز نخواهد توانست از طریق تعامل با محیط به سیاست تصمیم‌گیری مؤثری دست پیدا کند. همچنین، یادگیری تقویتی در ذات خود برای مسائل تعاملی و دارای بازخورد مرحله‌ای مناسب‌تر است، در حالی که این پروژه بیشتر جنبه طبقه‌بندی گسسته با داده‌های ایستا دارد.

فصل دهم:

جمع‌بندی و نتیجه‌گیری

در این فصل، پس از مرور و تحلیل نتایج به‌دست آمده از روش‌های مختلف تشخیص و دسته‌بندی خطا در سیستم‌هایAUV ، جمع‌بندی کلی درباره عملکرد هر روش و نقاط قوت و ضعف آنها ارائه می‌شود. این بررسی جامع، زمینه‌ای برای انتخاب بهترین روش یا ترکیبی از روش‌ها برای کاربردهای عملی فراهم می‌آورد.

ارزیابی هر روش

1. روش مبتنی بر مدل(Model-based)

تحلیل نتایج روش مبتنی بر مدل نشان می‌دهد که این رویکرد در کنترل ورودی‌ها و تخمین خطاها عملکرد قابل قبولی دارد. در محورهایX ، Yو عمق(Z) ، پاسخ سیستم به مسیر مرجع بسیار نزدیک بوده اما در بعضی نقاط، به‌خصوص در انتهای بازه زمانی، انحرافاتی دیده می‌شود. نقطه ضعف اصلی در محور Yaw است که ناپایداری و انحراف شدید مشاهده شده و نیازمند بهبود قابل توجه در بخش کنترل و تخمین است.

پیشنهادات بهبود شامل استفاده از کنترل‌کننده‌های غیرخطی مانند LQR غیرخطی، کنترل مقاوم و کنترل تطبیقی، بهبود ساختار شبکه‌های تخمین‌گر با استفاده از LSTM یاGRU ، افزایش کیفیت و تنوع داده‌های آموزشی، و به‌کارگیری فیلترهای کالمن پیشرفته (EKF) برای کاهش نویز تخمین‌ها است. همچنین مدلسازی دقیق‌تر اغتشاشات محیطی و پارامترهای دینامیکی غیرخطی می‌تواند به افزایش دقت مدل کمک کند.

2. روشSVM

روش ماشین بردار پشتیبان (SVM) عملکرد نسبتاً ضعیفی در تشخیص انواع خطاها نشان داد؛ با دقت کلی حدود 41٪ که نشان می‌دهد این مدل به دلیل پیچیدگی بالای داده‌ها و تشابه بین کلاس‌ها، در تمایز دقیق نمونه‌ها چندان موفق نبوده است. دقت پایین به ویژه در کلاس‌هایی مانند PropellerDamage_badنمایان است که نشان‌دهنده ناتوانی مدل در تشخیص حالات بحرانی است.

در نتیجه، SVMبرای این مسئله چندکلاسه و پیچیده چندان مناسب نیست و برای بهبود، نیاز به استفاده از ویژگی‌های دقیق‌تر یا روش‌های پیچیده‌تر است.

3. روش Deep Belief Network (DBN)

DBNتوانسته است عملکرد بسیار موفقی در تشخیص حالات مختلف خطا داشته باشد. دقت بیش از 95 درصد در تمامی کلاس‌ها، به‌ویژه دقت نزدیک به 100 درصد در کلاس‌هایی مانند AddWeight و PressureGain_constant، نشان‌دهنده قدرت بالای مدل در یادگیری ویژگی‌های پیچیده و تفکیک حالات مشابه است. این عملکرد قوی بیانگر این است که DBN می‌تواند به عنوان یک مدل قوی و قابل اعتماد برای سیستم‌های تشخیص عیب مورد استفاده قرار گیرد.

4. روش Autoencoder (MLP و(VAE

در بررسی دو مدل Autoencoder شامل MLP وVariational Autoencoder ، هر دو عملکرد قابل قبولی در تشخیص ناهنجاری‌ها ارائه دادند. دقت بالا (0.99) در هر دو مدل نشان می‌دهد که مدل‌ها توانایی شناسایی دقیق ناهنجاری‌ها را دارند. با این حال، مدل MLP در بازخوانی ناهنجاری‌ها (Recall) کمی بهتر عمل کرده (0.80 در برابر 0.75).

VAEبه دلیل بهره‌گیری از بازسازی داده‌ها و تحلیل خطای بازسازی، قابلیت‌های منحصر به فردی برای تشخیص ناهنجاری فراهم می‌کند و می‌تواند به طور تصویری به تحلیل خطا کمک کند. انتخاب بین این دو مدل بستگی به نیاز کاربرد و پیچیدگی داده‌ها دارد؛ MLP برای داده‌های ساده‌تر و VAE برای داده‌های پیچیده‌تر و تحلیلی مناسب‌تر است.

5. روش CNN

شبکه عصبی کانولوشنی (CNN) در این پروژه عملکرد بسیار برجسته‌ای داشت، با دقت کلی نزدیک به 98٪ و مقادیر بالا درprecision ، recallو f1-score برای تمامی کلاس‌ها. این نشان‌دهنده توانایی فوق‌العاده CNN در استخراج ویژگی‌های مؤثر و تشخیص دقیق حالات مختلف خطا است.

بنابراین، CNNیکی از بهترین گزینه‌ها برای کاربردهای عملی در تشخیص خطاهای AUV محسوب می‌شود.

6. ساختارResNet

ساختار ResNet با داده‌های مورد استفاده، عملکرد نسبتا ضعیفی داشت؛ با دقت کلی حدود 42٪ و مقادیر پایین در precision و .recall این نتیجه نشان می‌دهد که استفاده از ResNet در این مسئله خاص و با داده‌های موجود نیازمند بهینه‌سازی‌های بیشتر یا تغییر در پیش‌پردازش داده‌ها است.

7. یادگیری انتقالی(Transfer Learning)

روش‌های یادگیری انتقالی در این پروژه نتایج متفاوتی داشتند؛ مدل‌های یادگیری انتقالی روی داده‌های تصویری مانند VGG16 عملکرد نسبتا مناسبی داشتند (دقت 63٪)، اما روی داده‌های جدولی (Tabular)یا مدل‌های ResNet1D عملکرد ضعیفی ثبت شد. به طور کلی، یادگیری انتقالی روی داده‌های ساختاریافته جدولی چندان مؤثر نبوده و احتمالاً به دلیل تفاوت زیاد داده‌های مبدا و مقصد است.

8. یادگیری تقویتی(RL)

در این پروژه، بخش مربوط به یادگیری تقویتی تازه نوشته شده و نتایج آن به تفصیل ارائه نشده است، اما روش‌های RL به دلیل توانایی یادگیری سیاست‌های کنترلی و تصمیم‌گیری در شرایط پیچیده، می‌توانند در آینده نقش مهمی در بهبود تشخیص و کنترل سیستم‌های AUV ایفا کنند.

روش	دقت تقریبی(%)	مزایا	معایب	هزینه پیاده‌سازی و اجرا
مدل مبتنی بر تحلیل سیستم(Model-Based)	85-90	تحلیل فیزیکی و تبیین رفتار سیستم، پایه قوی کنترل	نیازمند مدل دقیق، حساس به پارامترها، پیچیدگی طراحی	متوسط (نیاز به مهندس سیستم و مدل‌سازی)
ماشین بردار پشتیبان(SVM)	~41	ساده و سریع، مناسب مسائل با داده‌های کم	دقت پایین در داده‌های پیچیده، عدم مقیاس‌پذیری بالا	کم (کتابخانه‌های آماده، اما نیاز به تنظیمات)
شبکه باور عمیق(DBN)	95+	دقت بسیار بالا، یادگیری ویژگی‌های پیچیده	نیازمند داده زیاد و زمان آموزش طولانی	زیاد (نیاز به منابع سخت‌افزاری قوی و زمان آموزش)
Autoencoder (MLP)	99	دقت بسیار بالا، شناسایی ناهنجاری‌های ظریف	نیاز به تنظیم دقیق، حساس به داده‌های آموزش	متوسط تا زیاد (بسته به پیچیدگی مدل)
Autoencoder (VAE)	99	قابلیت بازسازی و تحلیل خطا، شناسایی دقیق ناهنجاری	کمی پیچیده‌تر و زمان آموزش بیشتر نسبت به MLP	زیاد (پیچیده‌تر و نیازمند منابع)
شبکه عصبی کانولوشنی(CNN)	98	قابلیت استفاده از مدل‌های از پیش آموزش دیده	عملکرد ضعیف در این پروژه، نیاز به تنظیمات دقیق	زیاد (پیچیده و زمان‌بر)
یادگیری انتقالی	70-80	کاهش زمان آموزش، استفاده از مدل‌های قدرتمند	دقت متوسط، نیازمند داده‌های مشابه حوزه هدف	متوسط تا زیاد
یادگیری تقویتی(RL)	25	خودسازمان‌دهی و بهبود مستمر سیاست‌ها	نیاز به آموزش طولانی، پیچیدگی بالا	زیاد (نیاز به محاسبات و آموزش مکرر)

جدول 10-1 مقایسه روش های عیب یابی

نتیجه‌گیری کلی

با توجه به نتایج به‌دست آمده، می‌توان چنین جمع‌بندی کرد که:

- مدل‌های یادگیری عمیق، به‌ویژه شبکه‌های کانولوشنی (CNN) و شبکه‌های باور عمیق (DBN)، بهترین عملکرد را در تشخیص دقیق و پایدار انواع خطاها و ناهنجاری‌ها دارند و گزینه‌های بسیار مناسبی برای کاربردهای صنعتی و عملیاتی محسوب می‌شوند.
- روش‌های مبتنی بر مدل (Model-Based) با وجود محدودیت‌هایی، به دلیل قابلیت تبیین و تحلیل فیزیکی سیستم، همچنان پایه و اساس خوبی برای فهم رفتار سیستم و طراحی کنترل بهینه فراهم می‌کنند.
- الگوریتم‌های کلاسیک‌تر مانند SVM و یادگیری انتقالی در برخی کاربردها محدودیت‌هایی داشتند و نیاز به بهبود و تطبیق دقیق‌تر دارند.
- روش‌های نوین مانند Autoencoder ها با قابلیت بازسازی داده‌ها و تحلیل خطای بازسازی، به عنوان رویکردهای کمکی و مکمل می‌توانند در تحلیل‌های دقیق‌تر و شناسایی ناهنجاری‌های پیچیده‌تر بسیار موثر باشند.
- رویکردهای نوظهور یادگیری تقویتی پتانسیل بالایی برای آینده این حوزه دارند و باید به طور گسترده‌تری توسعه و آزمایش شوند.

در نهایت، انتخاب روش مناسب تشخیص خطا و ناهنجاری وابسته به شرایط عملیاتی، نوع داده‌ها، پیچیدگی سیستم و هدف کاربرد است و می‌توان از ترکیب چندین روش برای دستیابی به بهترین عملکرد بهره برد.

منابع و مراجع

[1] Autonomous underwater vehicle fault diagnosis dataset Daxiong Ji, Xin Yao, Shuo Li , Yuangui Tang , Yu Tian