****

**دانشکدگان علوم و فناوری های نوین**

**گروه بین رشته‌ای فناوری، مکاترونیک**

**کنترل همکارانه سیستم چند رباته چرخدار مبتنی بر یادگیری با در نظر گرفتن قیود متغیرهای حالت و ورودی کنترلی**

**نام دانشجو:**

**علی صفرزاده**

**استادان راهنما:   
 دکتر بهرام تارویردی‌زاده**

**دکتر خلیل عالی‌پور**

**استاد مشاور:  
 دکتر آرش قمری**

**پایان نامه برای دریافت درجه کارشناسی ارشد  
 در رشته مهندسی مکاترونیک**

**دی ماه 1402**

****

****

**دانشکده علوم و فنون نوین**

**گروه بین رشته‌ای فناوری، مکاترونیک**

**کنترل همکارانه سیستم چند رباته چرخدار مبتنی بر یادگیری با در نظر گرفتن قیود متغیرهای حالت و ورودی کنترلی**

**نگارنده:  
علی صفرزاده**

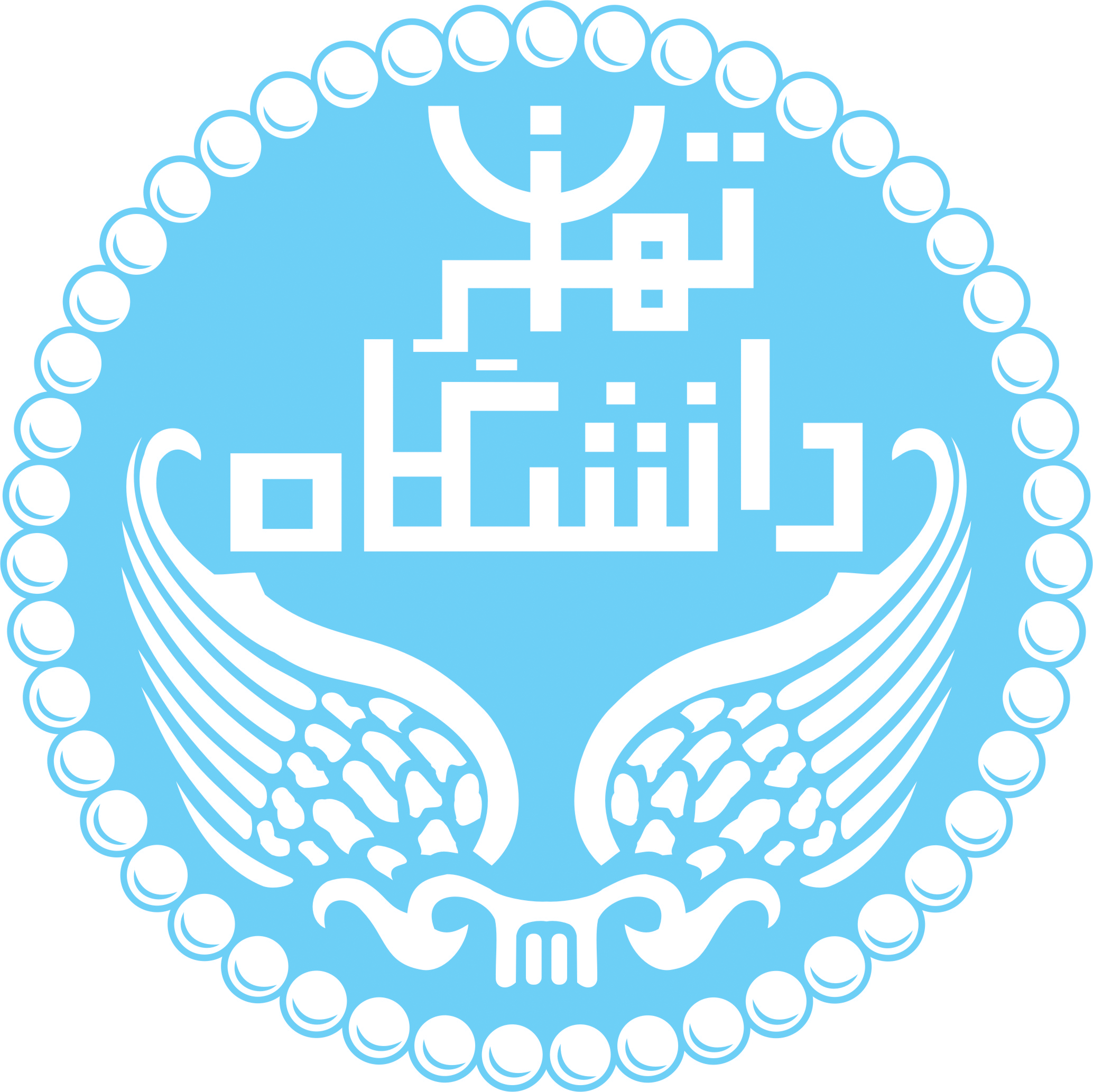
**استادان راهنما:   
دکتر بهرام تارویردی‌زاده**

**دکتر خلیل عالی‌پور**

**استاد مشاور:  
دکتر آرش قمری**

**پایان نامه برای دریافت درجه کارشناسی ارشد  
در رشته مهندسی مکاترونیک**

**بهمن ماه 1402**



**دانشگاه تهران**

**دانشکده علوم و فنون نوين**

**گروه** **بین رشته‌ای فناوری، مکاترونیک**

گواهي دفاع از پايان‌نامه كارشناسي ارشد

هيأت داوران پايان‌نامه كارشناسي ارشد ناپيوسته یا رساله دکتری

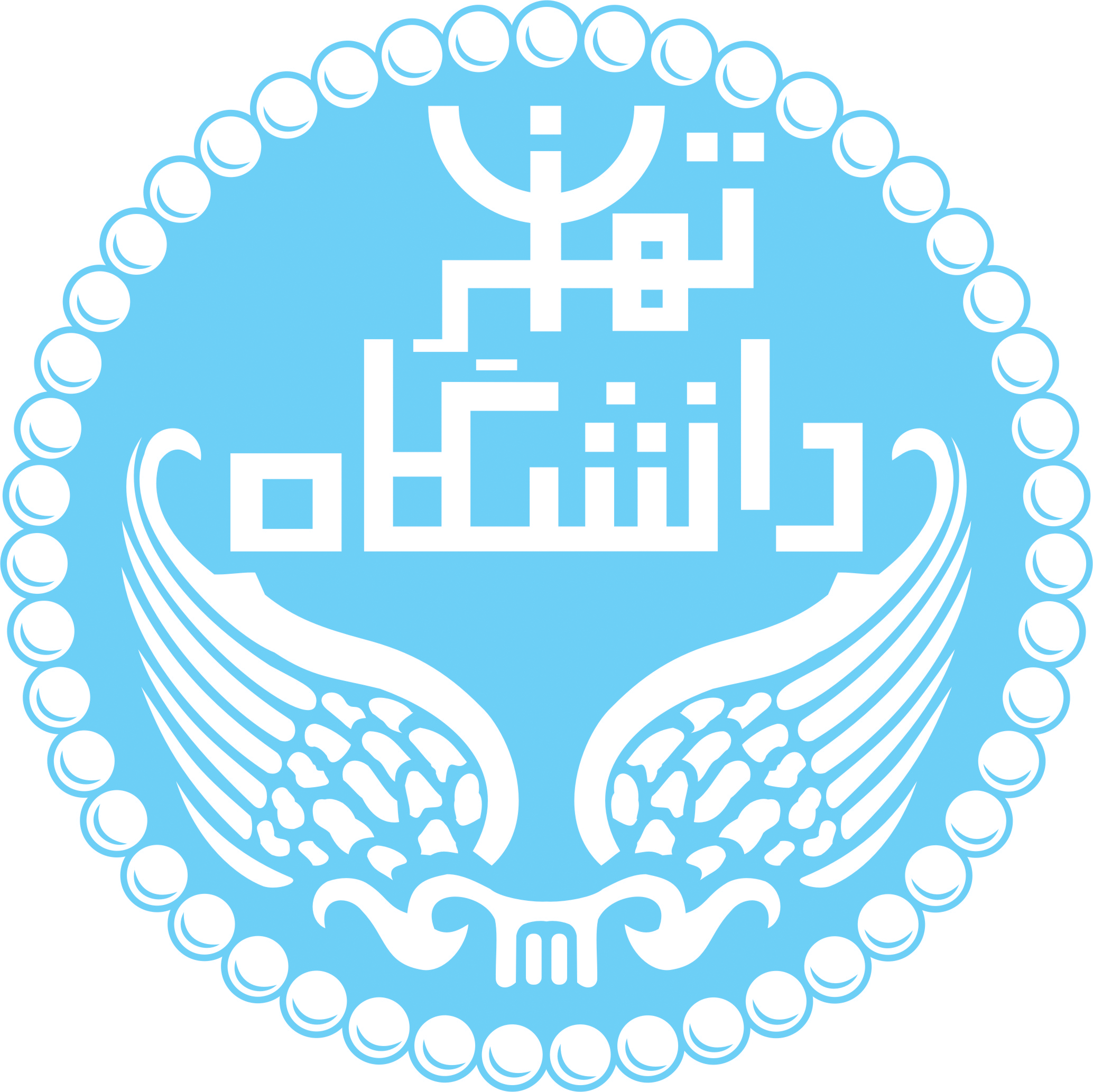
در رشته: **مهندسی مکاترونیک** گرايش:

با عنوان " **کنترل همکارانه سیستم چند رباته چرخدار مبتنی بر یادگیری با در نظر گرفتن قیود متغیرهای حالت و ورودی کنترلی**"

را در تاريخ با نمره نهايي به عدد به حروف

و درجه: ارزيابي نمود.

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **ردیف** | **عنوان هيأت داوران** | **نام و نام خانوادگي** | **مرتبه دانشگاهي** | **دانشگاه يا مؤسسه** | **امضاء** |
| 1 | استاد راهنما |  |  |  |  |
| 2 | استاد مشاور |  |  |  |  |
| 3 | استاد داور خارجی |  |  |  |  |
| 4 | استاد داور داخلی و نماينده كميته تحصيلات تكميلي |  |  |  |  |



**تعهد نامه اصالت اثر**

اینجانب علی صفرزاده دانش آموخته مقطع کارشناسی ارشد در رشتۀ مهندسی مکاترونیک که در تاریخ ........................... از پایان‌نامه یا رساله خود تحت عنوان: « کنترل همکارانه سیستم چند رباته چرخدار مبتنی بر یادگیری با در نظر گرفتن قیود متغیرهای حالت و ورودی کنترلی» با کسب درجۀ کارشناسي ارشد دفاع نموده‌ام، شرعاً و قانوناً متعهد می‌شوم:

1. مطالب مندرج در این پایان‌نامه یا رساله حاصل تحقیق و پژوهش اینجانب بوده و در مواردی که از دستاوردهای علمی‌و پژوهشی دیگران اعم از پایان‌نامه، کتاب، مقاله و غیره استفاده نموده‌ام، رعایت کامل امانت را نموده، مطابق مقررات، ارجاع و در فهرست منابع و مآخذ اقدام به ذكر آنها نموده‌ام.
2. تمامی ‌یا بخشی از این پایان‌نامه یا رساله قبلاً برای دریافت هیچ مدرک تحصیلی (هم سطح، پایین تر یا بالاتر) در سایر دانشگاه‌ها و مؤسسات آموزش عالی ارائه نشده است.
3. مقالات مستخرج از اين پایان‌نامه یا رساله کاملا حاصل کار اینجانب بوده و از هر گونه جعل داده و یا تغییر اطلاعات پرهیز نموده‌ام.
4. از ارسال همزمان و یا تکراری مقالات مستخرج از اين پایان‌نامه یا رساله (با بیش از 30 درصد همپوشانی) به نشریات و یا کنگره‌های گوناگون خودداری نموده و مي­نمایم.
5. کلیه حقوق مادّی و معنوی حاصل از این پایان‌نامه یا رساله متعلق به دانشگاه تهران بوده و متعهد می‌شوم هر گونه بهره مندی و یا نشر دستاوردهای حاصل از این تحقیق اعم از چاپ کتاب، مقاله، ثبت اختراع و غیره (چه در زمان دانشجویی و یا بعد از فراغت از تحصیل) با کسب اجازه از تیم استادان راهنما و مشاور و حوزه پژوهشی دانشکده باشد.

در صورت اثبات تخلف (در هر زمان) مدرك تحصيلي صادر شده توسط دانشگاه تهران از درجه اعتبار ساقط و اينجانب هیچگونه ادعایی نخواهم داشت.

امضا و نام و نام خانوادگی دانشجو: علی صفرزاده



تقدیم به پدر و مادر عزیزم که در فراز و نشیب زندگی، همواره پشتیبان و همراه من بوده‌اند.

**با تشکر و تقدیر از:**

**اساتید گرانقدرم، جناب آقای دکتر تارویردی‌زاده و جناب آقای دکتر خلیل عالی‌پور، که در مسیر اعتلای این تحقیق، بنده را بی دریغ یاری رساندند. امید که در تمامی مراحل زندگی در سایه سار لطف پروردگار باشند.**

**چکیده**

نقش ربات‌ها در زندگی انسان،‌ در بیش از دو دهه‌ی اخیر بسیار مورد توجه قرار گرفته و در حال افزایش است. از همین روی، دسته بندی‌های متنوعی برای علم رباتیک وجود دارد. یکی از مهمترین آن‌ها،‌ ربات‌های چرخدار است. همچنین با افزایش نقش ربات‌ها در زندگی بشر، کنترل آن‌ها و چالش‌های آن نیز طبعاً مورد بحث و بررسی قرار می‌گیرد. در این مبحث، به مفهوم کنترل خودکار ربات‌ها برمی‌خوریم که خود شامل الگوریتم‌های متعدد و پیچیدگی خاص خود است. بنابراین کنترل ربات‌های چرخدار که موضوع بحث این تحقیق است بسیار حائز اهمیت است. از سویی دیگر،‌ وظایفی وجود دارند که نیازمند همکاری بیش از یک ربات، جهت نیل به هدف می ‌باشد. در اینجا به مساله‌ی کنترل همکارانه‌ی ربات‌ها می‌رسیم که به طور کلی، چگونگی همکاری ربات‌ها در انجام یک وظیفه و چالش‌های کنترلی آن‌ را شامل می‌شود. همانطور که گفته شد تا بحال الگوریتم‌های بسیاری جهت کنترل ربات چرخدار ارائه شده‌اند که هر یک دارای نواقصی هستند. در این تحقیق سعی شده است با در نظر گرفتن این نواقص،‌ راه حل جدیدی برای یک سیستم جدید چند رباته ارائه گردد و بر روی پلتفرم‌های واقعی پیاده سازی شود. ابتدا برای یک سیستم تک پلتفرم مدل سینماتیکی و دینامیکی استخراج می‌شود. سپس مدل بدست آمده مورد شبیه سازی و تست قرار میگیرد. می‌دانیم که در این مدل عدم قطعیت وجود دارد. برای رسیدن به مدل دقیق تر و حل مشکل عدم قطعیت، با استفاده از شبکه عصبی RBF-NN به عنوان یک شبکه عصبی تخمینگر، مدل دینامیکی ربات در حال حرکت تخمین زده می‌شود و وزن‌های شبکه عصبی بهبود می‌یابند. حال این الگوریتم را برای سیستم چند رباته با ربات‌های همگن و یکسان تعمیم میدهیم و وزن‌های شبکه عصبی ربات‌ها بین آن‌ها به اشتراک گذاشته و رد و بدل می‌شوند. برای تک تک ربات‌های سیستم مسیر زمانی مرجع خاصی در نظر گرفته می‌شود، سپس با ارائه‌ی الگوریتم مناسب کنترلی همکارانه، و اعمال الگوریتم به سیستم، عمل دنبال کردن مسیر را بررسی می‌کنیم. برای اعمال الگوریتم کنترل همکارانه به سیستم چند رباته،‌ نیاز به تهیه و تولید پلتفرم‌های ربات چرخدار به تعداد ۳ عدد هست. بنابراین پس از ساخت پلتفرم‌ها، تمام الگوریتم را بر روی ۳ پلتفرم واقعی پیاده سازی و تست کرده، و نتایج بررسی می‌شوند.

**کلمات کلیدی:** سیستم چند رباته، شبکه عصبی تخمینگر، الگوریتم کنترلی همکارانه، پلتفرم ربات چرخدار، مدل دینامیکی غیر قطعی

**فهرست مطالب**

[1- فصل اول 7](#_Toc144741983)

[1˗1˗ مقدمه 8](#_Toc144741984)

[1˗2˗ مسئله تحقیق 9](#_Toc144741985)

[1˗3˗ تعریف موضوع تحقیق 10](#_Toc144741986)

[1˗4˗ اهداف کلی تحقیق 11](#_Toc144741987)

[1˗5˗ روش انجام تحقیق 12](#_Toc144741988)

[1˗6˗ چالش‌های تحقیق 13](#_Toc144741989)

[1˗7˗ نوآوری و اهمیت تحقیق 13](#_Toc144741990)

[1˗8˗ خلاصه فصل‌ها 15](#_Toc144741991)

[2- فصل دوم 16](#_Toc144741992)

[2˗1˗ مقدمه 17](#_Toc144741993)

[2˗2˗ تعاریف، اصول و مبانی نظری 17](#_Toc144741994)

[2˗2˗1˗ برگ گیاهان 17](#_Toc144741995)

[2˗2˗2˗ بیماری‌های مرتبط با برگ گیاهان و روش‌های مدیریت و مقابله با آن‌ها 18](#_Toc144741996)

[2˗2˗2˗1˗ زخم سیب 18](#_Toc144741997)

[2˗2˗2˗2˗ لکه برگی چشم غورباقه‌ای 19](#_Toc144741998)

[2˗2˗2˗3˗ زنگار 20](#_Toc144741999)

[2˗2˗2˗4˗ کپک پودری 21](#_Toc144742000)

[2˗3˗ مروری بر هوش مصنوعی 22](#_Toc144742001)

[2˗4˗ پردازش تصویر و کاربرد آن در مسائل کشاورزی 23](#_Toc144742002)

[2˗4˗1˗ فرایند‌های پردازش تصویر 23](#_Toc144742003)

[2˗5˗ یادگیری عمیق و کاربرد آن در مسائل کشاورزی 24](#_Toc144742004)

[2˗6˗ مروری بر ادبیات موضوع 25](#_Toc144742005)

[2˗6˗1˗ مجموعه‌داده‌های موجود شامل تصاویر بیماری برگ گیاهان 25](#_Toc144742006)

[2˗6˗2˗ مروری بر پیشینه تحقیق 27](#_Toc144742007)

[2˗7˗ نتیجه گیری 38](#_Toc144742008)

[3- فصل سوم 39](#_Toc144742009)

[3˗1˗ مقدمه 40](#_Toc144742010)

[3˗2˗ یادگیری ماشین 40](#_Toc144742011)

[3˗3˗ شبکه‌های عصبی 41](#_Toc144742012)

[3˗3˗1˗ روش پیش‌بینی شبکه 41](#_Toc144742013)

[3˗3˗2˗ روش آموزش پس‌انتشار 42](#_Toc144742014)

[3˗4˗ شبکه‌های عصبی پیچشی 43](#_Toc144742015)

[3˗4˗1˗ لایه پیچشی 44](#_Toc144742016)

[3˗4˗2˗ لایه تابع فعالسازی 45](#_Toc144742017)

[3˗4˗2˗1˗ تابع فعالسازی یکسوساز 46](#_Toc144742018)

[3˗4˗2˗2˗ تابع فعالسازی سیگموید 46](#_Toc144742019)

[3˗4˗2˗3˗ تابع فعالسازی سوویش 47](#_Toc144742020)

[3˗4˗2˗4˗ تابع فعالسازی بیشینه هموار 47](#_Toc144742021)

[3˗4˗3˗ لایه پَدینگ 48](#_Toc144742022)

[3˗4˗4˗ لایه ادغام 48](#_Toc144742023)

[3˗4˗5˗ لایه نرمال‌سازی دسته 49](#_Toc144742024)

[3˗4˗6˗ لایه برون‌انداز 49](#_Toc144742025)

[3˗4˗6˗1˗ لایه بلوک برون‌انداز 50](#_Toc144742026)

[3˗4˗7˗ لایه کاملا متصل 50](#_Toc144742027)

[3˗4˗8˗ تابع فعالسازی آخرین لایه 51](#_Toc144742028)

[3˗5˗ معماری‌های شبکه‌های عصبی پیچشی 51](#_Toc144742029)

[3˗5˗1˗ معماری Resnet و Resnext 51](#_Toc144742030)

[3˗5˗2˗ معماری Efficientnet 53](#_Toc144742031)

[3˗5˗3˗ معماری SWin 55](#_Toc144742032)

[3˗5˗4˗ معماری Regnet 56](#_Toc144742033)

[3˗5˗5˗ معماری ConvNext 58](#_Toc144742034)

[3˗5˗6˗ معماری WideResnet 60](#_Toc144742035)

[3˗6˗ یادگیری انتقالی 61](#_Toc144742036)

[3˗7˗ الگوریتم‌های بهنیه سازی ازدحامی 62](#_Toc144742037)

[3˗8˗ نرم‌افزار و سخت‌افزارهای لازم برای آموزش شبکه عصبی 63](#_Toc144742038)

[3˗8˗1˗ CPU 63](#_Toc144742039)

[3˗8˗2˗ GPU 63](#_Toc144742040)

[3˗8˗3˗ پایتون 64](#_Toc144742041)

[3˗9˗ شاخص عملکرد 64](#_Toc144742042)

[4- فصل چهارم 66](#_Toc144742043)

[4˗1˗ مقدمه 67](#_Toc144742044)

[4˗2˗ مجموعه‌داده 67](#_Toc144742045)

[4˗3˗ توسعه و روند آموزش مدل شبکه عصبی 70](#_Toc144742046)

[4˗3˗1˗ مشخصات سخت‌افزار و نرم‌افزار جهت پیاده‌سازی شبکه 71](#_Toc144742047)

[4˗3˗2˗ پلتفرم Pytorch 71](#_Toc144742048)

[4˗3˗3˗ تقسیم بندی داده‌ها 71](#_Toc144742049)

[4˗3˗4˗ تقسیم‌بندی داد‌های آموزشی 72](#_Toc144742050)

[4˗3˗5˗ مدل‌های شبکه‌ی عصبی پیچشی 72](#_Toc144742051)

[4˗3˗6˗ نرم‌سازی برچسب‌ها 72](#_Toc144742052)

[4˗3˗7˗ تابع هزینه 73](#_Toc144742053)

[4˗3˗8˗ الگوریتم بهینه‌سازی 74](#_Toc144742054)

[4˗3˗9˗ نرخ یادگیری 74](#_Toc144742055)

[4˗3˗10˗ سایر ابرپارامتر‌های آموزش شبگه 74](#_Toc144742056)

[4˗4˗ آزمون‌ها و نتایج 74](#_Toc144742057)

[4˗4˗1˗ آزمون اول، آموزش و ارزیابی شبکه‌های عصبی پیچشی 75](#_Toc144742058)

[4˗4˗1˗1˗ نتایج مدل EfficientnetV2S در آزمون اول 76](#_Toc144742059)

[4˗4˗1˗2˗ نتایج مدل EfficientnetV2M در آزمون اول 77](#_Toc144742060)

[4˗4˗1˗3˗ نتایج مدل Resnet50 در آزمون اول 78](#_Toc144742061)

[4˗4˗1˗4˗ نتایج مدل Resnext5032x4D در آزمون اول 79](#_Toc144742062)

[4˗4˗1˗5˗ نتایج مدل WideResnet50 در آزمون اول 80](#_Toc144742063)

[4˗4˗1˗6˗ نتایج مدل RegnetY8GF در آزمون اول 81](#_Toc144742064)

[4˗4˗1˗7˗ نتایج مدل SwinV2T در آزمون اول 82](#_Toc144742065)

[4˗4˗1˗8˗ نتایج مدل SwinV2S در آزمون اول 83](#_Toc144742066)

[4˗4˗1˗9˗ نتایج مدل ConvNextT در آزمون اول 84](#_Toc144742067)

[4˗4˗1˗10˗ نتایج مدل ConvNextS در آزمون اول 85](#_Toc144742068)

[4˗4˗2˗ آزمون دوم، بهینه‌سازی آستانه‌ی برچسب‌ها 86](#_Toc144742069)

[4˗5˗ نتیجه گیری 88](#_Toc144742070)

[5- فصل پنجم 89](#_Toc144742071)

[5˗1˗ مقدمه 90](#_Toc144742072)

[5˗2˗ محتوا 90](#_Toc144742073)

[5˗2˗1˗ تکنیک‌های استفاده شده دیگر 90](#_Toc144742074)

[5˗2˗2˗ جمع‌بندی 91](#_Toc144742075)

[5˗3˗ نوآوری 92](#_Toc144742076)

[5˗3˗1˗ محدودیت‌ها 93](#_Toc144742077)

[5˗3˗2˗ پیشنهادها 94](#_Toc144742078)

[6- مراجع 95](#_Toc144742079)

[7- واژه نامه 105](#_Toc144742080)

**فهرست شکل‌ها**

[شکل ‏2-1- لایه‌های اصلی موجود در اکثر برگها 18](#_Toc144742433)

[شکل ‏2-2- تصاویر نمونه‌ای از بیماری زخم سیب بر روی برگها 19](#_Toc144742434)

[شکل ‏2-3- تصویر نمونه‌ای از بیماری لکه برگی چشم غورباقه‌ای بر روی برگها 20](#_Toc144742435)

[شکل ‏2-4- تصویر نمونه‌ای از بیماری زنگار الف) سطح بر روی برگ و ب) نشانه بیماری در زیر برگ [46] 21](#_Toc144742436)

[شکل ‏2-5- تصویر نمونه‌ای از بیماری کپک پودری بر روی برگها 22](#_Toc144742437)

[شکل ‏2-6- تصویر نمونه‌ای از طبقه‌های مجموعه‌داده Plant Village [7] 26](#_Toc144742438)

[شکل ‏2-7- تصویر نمونه‌ای از طبقه‌های مجموعه‌داده Plant Pathology 2020 [24] 27](#_Toc144742439)

[شکل ‏2-8- نمونه‌ای از انواع تصاویر استفاده شده در تحقیق موهانتی و همکاران الف) بدون پس‌زمینه ب) حالت سیاه و سفید و ج) تصاویر عادی [7] 29](#_Toc144742440)

[شکل ‏2-9- تصاویر نمونه‌ای از خروجی قسمت طبقه‌بند بر پایه خوشه‌بندی و طبقه‌بندی بایزی الگوریتم ارائه شده توسط ژوهانس و همکاران [75] 29](#_Toc144742441)

[شکل ‏2-10- مقایسه نمونه عکس‌های استفاده شده در تحقیق فرنتینوس؛ نمونه عکس‌های گرفته شده در الف) شرایط واقعی و محیط کشاورزی و ب) عکس‌های گرفته شده در محیط آزمایشگاه [23] 30](#_Toc144742442)

[شکل ‏2-11- نمودارهای دقت و هزینه مربوط به شبکه Densenet121 در تحقیق تو و همکاران [75] 31](#_Toc144742443)

[شکل ‏2-12- نمونه‌ای از تصاویر استفاده شده در تحقیق باربدو. الف) مجموعه‌ای از بافت‌های بیماری ب) بافت برگ جدا شده و ج) علائم کوچک و پراکنده بیماری [25] 32](#_Toc144742444)

[شکل ‏2-13- نمودار دقت و هزینه در طول فرایند آموزش مدل در تحقیق سمباسیوَم و همکارش الف) نمودار دقت نسبت به دوره‌های آموزش ب) نمودار هزینه نسبت به دوره‌های آموزشی [79] 34](#_Toc144742445)

[شکل ‏2-14- عملیات انجام شده توسط لیو و همکاران برای طبقه‌بندی عکس‌ها [26] 35](#_Toc144742446)

[شکل ‏2-15- نمودار عملکرد مدل ارائه شده در تحقیق وُرا و همکاران. هر تصویر برگ به سه مدل داده شده و از آنها خروجی گرفته می‌شود و طبقه‌ای که بیشترن تعداد رأی را داشته باشد، به عنوان طبقه‌ی نهایی انتخاب می‌شود. در صورتی که تعداد آرای یکی بود، طبقه خروجی شبکه Xception انتخاب می‌شود. 35](#_Toc144742447)

[شکل ‏2-16- شبکه ارائه شده توسط یاداو و همکاران جهت طبقه‌بندی بیماری‌های قارچی برگ درخت سیب [81] 36](#_Toc144742448)

[شکل ‏2-17- نمودار الف) دقت و ب) هزینه مدل ارائه شده در تحقیق پرستیو و همکاران [84] 37](#_Toc144742449)

[شکل ‏3-1- ساختار یک شبکه عصبی با 2 لایه مخفی 41](#_Toc144742450)

[شکل ‏3-2- شبکه عصبی پرسپترون 42](#_Toc144742451)

[شکل ‏3-3- روش آموزش پس‌انتشار 43](#_Toc144742452)

[شکل ‏3-4- یک نمونه شبکه عصبی پیچشی برای طبقه بندی تصاویر مربوط به اعداد، شامل 2 لایه پیچشی، دو لایه ادغام بیشینه و دو لایه کاملا متصل 44](#_Toc144742453)

[شکل ‏3-5- نمونه‌ای از عملیات کانولوشن 45](#_Toc144742454)

[شکل ‏3-6- نمودار تابع فعالسازی الف) یکسوساز خطی و ب) سیگموید 47](#_Toc144742455)

[شکل ‏3-7- نمودار تابع فعالسازی الف) سوویش و ب) بیشنیه هموار برای 2 متغیر 48](#_Toc144742456)

[شکل ‏3-8- نمونه‌ای از کاربرد لایه Padding برای حفظ ابعاد تصویر 48](#_Toc144742457)

[شکل ‏3-9- نمونه‌ای از عملکرد لایه‌های ادغام بیشنیه و میانگین 49](#_Toc144742458)

[شکل ‏3-10- مقایسه عملکرد شبکه در حین آموزش الف) بدون لایه برون‌انداز ب) با لایه برون‌انداز 50](#_Toc144742459)

[شکل ‏3-11- مقایسه روش‌های برون‌اندازی و خروجی شده توسط الف) لایه برون‌انداز معمولی ب) لایه بلوک برون‌انداز [109] 51](#_Toc144742460)

[شکل ‏3-12- ساختار بلوک میانبر در معماری شبکه Resnet [90] 52](#_Toc144742461)

[شکل ‏3-13- مقایسه ساختارهای بلوک‌های سازنده الف) شبکه Resnext با کاردینالیته 32 و ب) شبکه Resnet با یک دیگر. هر دو بلوک پیچیدگی تقریبا یکسانی دارند و اتصالات میانبر در آنها رسم شده است [110] 52](#_Toc144742462)

[شکل ‏3-14- مقایسه‌ی بین ساختار شبکه‌های Resnet50 و Resnext50 در قالب 32x4d. میزان کاردینالیته=32 نشان‌دهنده تجمع کانولوشن‌ در 32 گروه است [110] 53](#_Toc144742463)

[شکل ‏3-15- روش مقیاس‌بندی در حین طراحی مدل‌های شبکه‌ی عصبی پیچشی. الف) مدل اصلی ب) افزایش عرض ج) افزایش عمق د) افزایش وضوح ورودی و ه) افزایش همه پارامتر‌ها به صورت ترکیبی [88] 54](#_Toc144742464)

[شکل ‏3-16- ساختار شبکه EfficientnetB0 [88] 55](#_Toc144742465)

[شکل ‏3-17- معماری شبکه ViT. تصاویر به چند بخش با وضوح ثابت تقسیم شده و هر کدام از آنها به صورت داده‌های خطی در می‌آیند و هر بخش تصویر در کنار موقعیت قرارگیری آن، به یک مبدل استاندارد داده می‌شوند [112] 56](#_Toc144742466)

[شکل ‏3-18- مقایسه عملکرد بین دو شبکه‌ی الف) ViT که مبدل ارائه شده نقشه ویژگی‌هایی با وضوح کمتر تولید می‌کند و ب) SwinT که مبدل آن، نقشه ویژگی‌ها به صورت سلسله‌مراتبی و با وضوح بالاتر می‌سازد [90] 56](#_Toc144742467)

[شکل ‏3-19- نمایش معماری شبکه Swin Transformer [90] 56](#_Toc144742468)

[شکل ‏3-20- ساختار کلی شبکه‌های طراحی شده RegNet الف) بلوک‌هایی موجود در مرحله i ب) مرحله‌های موجود در بدنه و ج) ساختار کلی شبکه اولیه معرفی شده [92] 57](#_Toc144742469)

[شکل ‏3-21-نقشه راه ارائه شده در [113] برای ارائه شبکه‌ی پیچشی جدید 59](#_Toc144742470)

[شکل ‏3-22- معماری شبکه ConvNext پایه (اتصال‌های میانبر نمایش داده نشده اند) 60](#_Toc144742471)

[شکل ‏3-23- مقایسه شبکه ConvNext با دیگر شبکه‌ها [113] 60](#_Toc144742472)

[شکل ‏3-24- نواع بلوک‌های الف) ساده ب) bottleneck ج) عریض و ه) عریض با لایه برون انداز استفاده شده در شبکه WideResnet. بعد از هر لایه کانولوشن لایه‌های نرمال‌سازی دسته و تابع فعالسازی یکسوساز قرار گرفته است [114] 61](#_Toc144742473)

[شکل ‏4-1- نمونه‌ای از بیماری‌های موجود در مجموعه‌داده مورد استفاده در این تحقیق: الف) لکه چشم غورباقه‌ای (frog eye leaf spot) ب) زخم سیم (scab) ج) کپک پودری (powdery mildew) د) زنگار (rust) ه) زنگار و و) نامشخص (complex) [27] 67](#_Toc144742474)

[شکل ‏4-2- نمودار فراوانی مربوط به هر طبقه موجود در مجموعه‌داده Plant Pathology 2021 68](#_Toc144742475)

[شکل ‏4-3- نمودار فراوانی درصدی مربوط به مجموعه‌داده Plant Pathology 2021 68](#_Toc144742476)

[شکل ‏4-4- نمودار فراوانی داده‌های Plant Pathology 2021 به صورت چندبرچسبه الف) بر اساس تعداد ب) بر اساس درصد داده‌‎ها 69](#_Toc144742477)

[شکل ‏4-5-فرایند آموزش با استفاده از تکنیک ارزیابی 5-قسمتی. رنگ سبز نشان‌دهنده دسته‌هایی است که برای فرایند آموزش استفاده می‌شوند و رنگ نارنجی نشان‌دهنده دسته‌هایی است که برای ارزیابی مدل‌ها در حین آموزش استفاده می‌شوند 72](#_Toc144742478)

[شکل ‏4-6- مقایسه تاثیر نرخ‌های مختلف نرم‌سازی برچسب‌ها بر روی داده‌های مجموعه‌داده Cifar-10 الف) نرخ نرم‌سازی صفر ب) نرخ نرم‌سازی 0.2 ج) نرخ نرم‌سازی 0.7 [115] 73](#_Toc144742479)

[شکل ‏4-7- نمودارهای حاصل شده در فرایند آموزش شبکه EfficientnetV2S الف) دقت ب) هزینه ج) امتیاز اف-1 76](#_Toc144742480)

[شکل ‏4-8- نمودارهای حاصل شده در فرایند آموزش شبکه EfficientnetV2M الف) دقت ب) هزینه ج) امتیاز اف-1 77](#_Toc144742481)

[شکل ‏4-9- نمودارهای حاصل شده در فرایند آموزش شبکه Resnet50 الف) دقت ب) هزینه ج) امتیاز اف-1 78](#_Toc144742482)

[شکل ‏4-10- نمودارهای حاصل شده در فرایند آموزش شبکه Resnext5032x4D الف) دقت ب) هزینه ج) امتیاز اف-1 79](#_Toc144742483)

[شکل ‏4-11- نمودارهای حاصل شده در فرایند آموزش شبکه Wideresnet50 الف) دقت ب) هزینه ج) امتیاز اف-1 80](#_Toc144742484)

[شکل ‏4-12- نمودارهای حاصل شده در فرایند آموزش شبکه RegnetY8GF الف) دقت ب) هزینه ج) امتیاز اف-1 81](#_Toc144742485)

[شکل ‏4-13- نمودارهای حاصل شده در فرایند آموزش شبکه SwinV2T الف) دقت ب) هزینه ج) امتیاز اف-1 82](#_Toc144742486)

[شکل ‏4-14- نمودارهای حاصل شده در فرایند آموزش شبکه SwinV2S الف) دقت ب) هزینه ج) امتیاز اف-1 83](#_Toc144742487)

[شکل ‏4-15- نمودارهای حاصل شده در فرایند آموزش شبکه ConvNextT الف) دقت ب) هزینه ج) امتیاز اف-1 84](#_Toc144742488)

[شکل ‏4-16- نمودارهای حاصل شده در فرایند آموزش شبکه ConvNextS الف) دقت ب) هزینه ج) امتیاز اف-1 85](#_Toc144742489)

[شکل ‏5-1- نمونه‌ای از تصویر برش-ترکیب شده به همراه برچسب‌های آن 91](#_Toc144742490)

[شکل ‏5-2- نمونه‌ای از تصاویر نامشخص موجود در مجموعه‌داده 93](#_Toc144742491)

**فهرست جداول**

[جدول ‏2-1- مقایسه تکنیک‌های داده‌افزایی در تحقیق نون و همکاران [76] 33](#_Toc144742551)

[جدول ‏2-2- نتایج به‌دست آمده از مقایسه بهینه‌سازهای متفاوت برای طبقه‌بندی بیماری‌های برگ گیاه [76] 33](#_Toc144742552)

[جدول ‏2-3- مقایسه نتایج حاصل شده از تحقیق لیو با مدل‌های رایج و روش آموزش معمولی بر روی مجموعه‌داده‌ی PDD271 [26]. 35](#_Toc144742553)

[جدول ‏2-4- نتایج حاصل از تحقیق پارِز و همکاران [83] 37](#_Toc144742554)

[جدول ‏3-1- مقایسه ساختار‌های معماری الف) RegnetX و ب) RegnetY با یکدیگر [92]. 58](#_Toc144742555)

[جدول ‏3-2- مقایسه بین مدل‌های از پیش‌آموزش دیده‌شده مورد استفاده در این تحقیق 62](#_Toc144742556)

[جدول ‏4-1- فراوانی‌داده‌های مجموعه‌داده Plant Pathology 2021 به صورت چند طبقه. 70](#_Toc144742557)

[جدول ‏4-2- فراوانی داده‌های مجموعه‌داده Plant Pathology 2021 پس از عملیات تبدیل طبقات اصلی به حالت چند برچسبه 70](#_Toc144742558)

[جدول ‏4-3- درصد شاخص‌های عملکرد مدل EfficientnetV2S برای هر برچسب 76](#_Toc144742559)

[جدول ‏4-4- درصد شاخص‌های عملکرد مدل EfficientnetV2M برای هر برچسب 77](#_Toc144742560)

[جدول ‏4-5- درصد شاخص‌های عملکرد مدل Resnet50 برای هر برچسب 78](#_Toc144742561)

[جدول ‏4-6- درصد شاخص‌های عملکرد مدل Resnext5032x4D برای هر برچسب 79](#_Toc144742562)

[جدول ‏4-7- درصد شاخص‌های عملکرد مدل Wideresnet50 برای هر برچسب 80](#_Toc144742563)

[جدول ‏4-8- درصد شاخص‌های عملکرد مدل RegnetY8GF برای هر برچسب 81](#_Toc144742564)

[جدول ‏4-9- درصد شاخص‌های عملکرد مدل SwinV2T برای هر برچسب 82](#_Toc144742565)

[جدول ‏4-10- درصد شاخص‌های عملکرد مدل SwinV2S برای هر برچسب 83](#_Toc144742566)

[جدول ‏4-11- درصد شاخص‌های عملکرد مدل ConvNextT برای هر برچسب 84](#_Toc144742567)

[جدول ‏4-12- درصد شاخص‌های عملکرد مدل ConvNextS برای هر برچسب 85](#_Toc144742568)

[جدول ‏4-13- نتایج به دست آمده پس از اعمال تغییرات یافته شده توسط الگوریتم‌های بهینه‌سازی ازدحامی 86](#_Toc144742569)

[جدول ‏4-14- مشخصات الگوریتم‌های بهینه‌سازی ازدحامی که در این تحقیق استفاده شده است 87](#_Toc144742570)

[جدول ‏4-15- رتبه بندی بهترین الگوریتم‌های بهینه‌سازی بر اساس دقت مدل‌ها 88](#_Toc144742571)

# فصل اول

کلیات تحقیق

## مقدمه

امروزه مبحث رباتیک و استفاده از ربات، برای بهبود بخشی به زندگی انسان و سهولت بیشتر انجام کارها، خصوصا مواردی که موجب آسیب فیزیکی به انسان میشود، یا انجام آن برای انسان دشوار است، بسیار مورد توجه و در حال پیشرفت و گسترده‌تر شدن می‌باشد. یکی از زیر مجموعه‌های مهم رباتیک، بخش ربات‌های چرخدار وکنترل آنهاست. در این تحقیق، مسأله، چگونگی کنترل یک سیستم چند رباته[[1]](#footnote-1) به صورت همکارانه[[2]](#footnote-2) می‌باشد به طوری که ربات‌ها بتوانند مسیر مورد نظر را با دقت مناسب دنبال کنند[[3]](#footnote-3). این همکاری توسط یک شبکه[[4]](#footnote-4) با جابجایی اطلاعات بین ربات‌ها تعریف می‌شود. می‌دانیم که در مدلسازی و اندازه‌گیری پارامترهای فیزیکی ربات عدم قطعیت[[5]](#footnote-5) وجود دارد. چرا که دقت اندازه گیری تابع مستقیمی از ابزار اندازه گیری و دقت شخص اندازه گیرنده است. بنابراین به طور کلی عدم قطعیت در مدل دینامیکی به معنای عدم دقت در اندازه گیری پارامترهای فیزیکی (مانند جرم ربات، ممان اینرسی،‌ شعاع چرخ‌ها و...) می‌باشد. فلذا برای کنترل بهتر ربات، باید کنترلر به گونه‌ای طراحی شود که بر این مشکل چیره شود. جهت تخمین دینامیک ربات و چیره‌گی بر عدم قطعیت در مدل سیستم (عدم دقت در اندازه گیری پارامترهای فیزیکی در مدلسازی دینامیکی) از شبکه عصبی RBF-NN که نوعاً یک شبکه عصبی تخمینگر است استفاده میشود. به این صورت که در حالت معمول، یک مدلسازی دینامیکی از ربات صورت می‌پذیرد که شامل پارامترهای فیزیکی است. سپس با اندازه گیری این پارامترها،‌ مدل دینامیکی به صورت عددی (و با عدم قطعیت) بیان می‌شود و مسلما کنترلر طراحی شده بر پایه‌ی این مدل دقیق نیست. اما در این روش پس از اندازه گیری پارامترها و بیان به صورت عددی، بخش اعظم مدل دینامیکی (شامل ماتریس جرم و ممان، ماتریس گشتاور گرانشی، ماتریس تبدیل سرعت و...) توسط RBF-NN تخمین زده میشود و مدل دینامیکی ربات را تکمیل می‌کند. ورودی این شبکه عصبی مکان، سرعت و شتاب لحظه‌ای ربات است که با فیدبک بدست می‌آیند و خروجی آن تکمیل مدل دینامیکی است[1].

هدف از انجام این تحقیق حصول و تولید یک کنترل کننده است که ربات‌ها در سیستم چند رباته بتوانند با دقت بالایی مسیر مورد نظر را دنبال کنند و این الگوریتم بر روی یک سیستم ۳ رباته واقعی پیاده سازی شود.

در دسته بندی ربات‌ها، می‌توان به سه دسته‌ی عمده اشاره نمود. ربات‌های پرنده، ربات‌های متحرک در آب، و ربات های پایه متحرک. که ربات‌های پایه متحرک را نیز به دو دسته‌ی کلی ربات‌های پایه متحرک دارای پا[[6]](#footnote-6) و ربات‌های پایه متحرک چرخ‌دار[[7]](#footnote-7)، می‌توان تقسیم نمود. از این میان، ربات‌های پایه متحرک چرخدار پرکاربردترین و از نگاهی ساده ‌ترین ربات‌ها هستند. این ربات‌ها مدل سینماتیکی و دینامیکی ساده‌تری دارند و همچنین برای کنترل کردن آن‌ها، با توجه به سلب بودن زمین، چالش‌های حرکتی در آب و در هوا برای آن‌ها مطرح نمی‌شود. و نیز مصرف انرژی برای جابجا شدن آنها کمینه‌است و معمولاً سرعت حرکت آنها از نوع‌های دیگر ربات‌های پایه متحرک بیشتر است.

## انواع ربات‌های پایه متحرک چرخدار

ربات‌های پایه متحرک چرخدار انواع مختلفی دارند و به تبع، مدلسازی آنها نیز متفاوت می‌شود. در این قسمت به تعریف و تشریح انواع زیر می‌پردازیم:

* ربات‌های یک چرخ
* ربات‌های دو چرخ
* ربات‌های سه چرخ
* ربات‌های چهار چرخ

### ربات‌های یک چرخ

به دلیل آنکه پایدار سازی و کنترل ربات‌های تک چرخ ساده نیست، این نوع ربات‌‌‌ها کم کاربردتر هستند. اما به لحاظ ساخت، مقرون به صرفه‌تر هستند و هزینه‌ی کمتری دارند. به دلیل آنکه نقطه‌ی تماس ربات تک چرخ با زمین، تنها یک نقطه است، حفظ تعادل، پایدار سازی و کنترل آن دشوار است. با این اوصاف استفاده از یک چرخ کروی برای حفظ تعادل و امکان حرکت در تمامی جهات، می‌تواند گزینه‌ی مناسبی باشد. با این حال رواج استفاده از چرخ استوانه‌ای و تایر بیشتر است. به این علت که حرکت دادن تایر توسط یک موتور بسیار ساده تر است. در شکل زیر یک نمونه ربات تک چرخ دیده می‌شود که تعادل خود را حفظ نموده است.

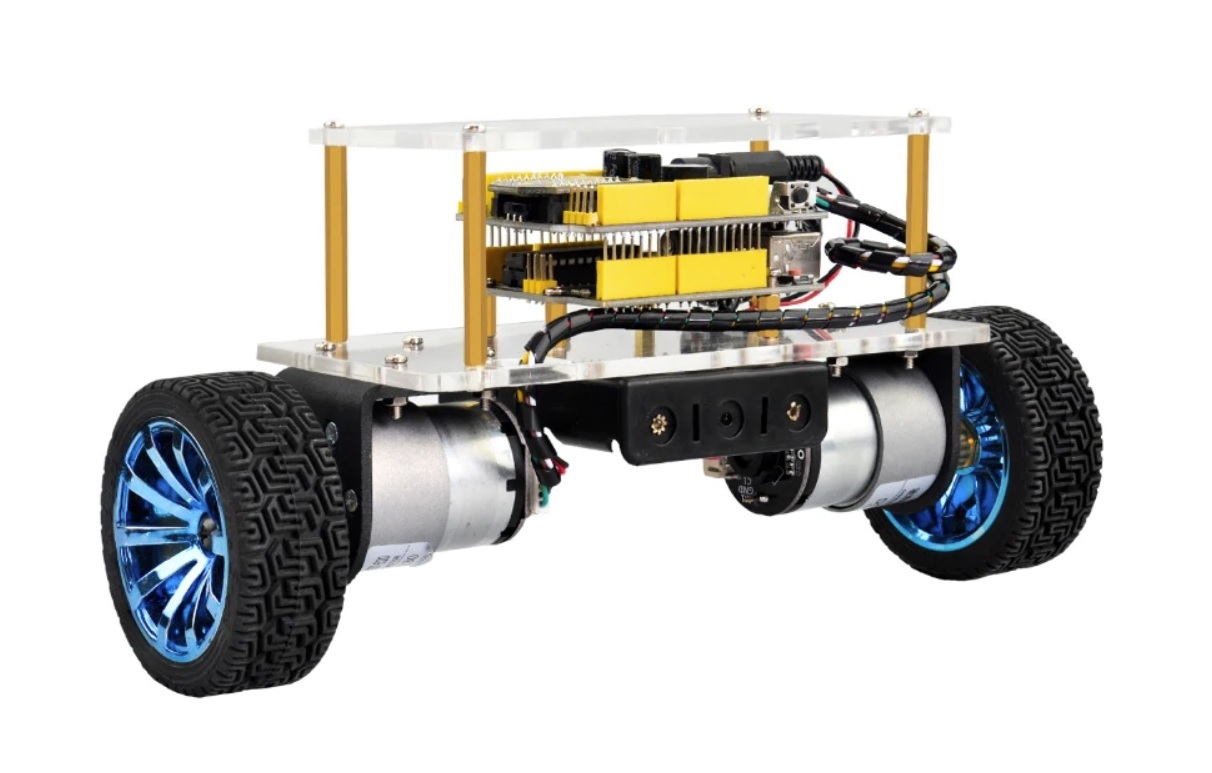


شکل 1 یک ربات تک چرخ Unicycle

### ربات‌های دو چرخ

در نگاه اول شاید کنترل ربات دو چرخ ساده به نظر بیاید. اما در عمل مبحث دشواری ست. ربات دو چرخ یک نوع پلتفرم با دو چرخ موازی زیر ربات است که مدار و متعلقات مکانیکی بر روی آن دو سوار هستند. برای حفظ تعادل و پایدار سازی ربات دو چرخ عموماً مرکز ثقل آنرا پایین و در یک نقطه نگه می‌دارند. برای این منظور معمولاُ باتری یا دیگر متعلقات با وزن بیشتر را پایین‌تر قرار می‌دهند. حرکت ربات دو چرخ، ساده است و برای حرکت مستقیم کافی ست هر دو چرخ با دور بر دقیقه[[8]](#footnote-8) یکسان حرکت کنند. و اگر یک چرخ با سرعت بیشتر از چرخ دیگر بچرخد، ربات حرکت دورانی در جهت مقابل اجام می‌دهد.

شکل زیر یک نمونه ربات دو چرخ را نمایش می‌دهد.



شکل 2 ربات دو چرخ ساده

### ربات‌های سه چرخ

ربات‌های سه چرخ شاید از رایج ترین انواع ربات‌های چرخدار باشند. چرا که سرعت حرکت این نوع در تمامی جهات از دو نوع قبلی بیشتر بوده، و همچنین حفظ تعادل آنها مساله‌ی چالش برانگیزی نیست. در این نوع ربات‌ها، دو دسته بندی وجود دارد. دسته‌ی اول ربات‌هایی هستند که هر به سه چرخ موتور متصل بوده و کار انقال قدرت صورت می‌پذیرد. مانند شکل زیر



شکل 3 ربات سه چرخ 3-wheel drive

نوع دیگر ربات‌های سه چرخ، که پلتفرم استفاده شده در این تحقیق نیز از همین مدل است، نوعی ست که دو چرخ متصل به موتور در عقب ربات قرار دارند. و یک چرخ هرزگرد که وظیفه‌ی حفظ تعادل ربات را بر عهده دارد، در جلوی ربات قرار دارد. مانند شکل زیر



شکل 4 ربات سه چرخ 2-wheel drive

### ربات‌های چهار چرخ

رواج ربات‌های چهار چرخ، از ربات‌های سه چرخ نیز بیشتر است. به لحاظ پایداری و کنترل پذیری، این ربات‌ها دارای ارجحیت هستند. همچنین برای تعیین مرکز ثقل ربات چهار چرخ چالش کمتری وجود دارد چرا که ربات در چهار نقطه روی زمین قرار دارد. از معایب ربات چهار چرخ اما می‌توان به بحث هزینه اشاره کرد. با توجه به اینکه ربات چهار چرخ دارای چهار چرخ و بعضاً چهار موتور می‌باشد، به لحاظ هزینه‌ی ساخت از ربات‌های ساده‌تر در رده‌ی پایین تری قرار می‌گیرد.

برای دسته بندی ربات‌های چهار چرخ از لحاظ کنترل چرخ‌ها و انتقال قدرت، می‌توان آنها را به سه دسته‌ی کلی تقسیم نمود.

دسته اول به این صورت است که برای انتقال توان، از هر ۴ چرخ استفاده می‌کنیم و همه‌ی چرخ‌ها را به موتور متصل می‌کنیم. در این حالت کنترل و جهت دهی به ربات به این صورت است که برای چرخش به سمت چپ، دور موتورهای راست جلو و راست عقب را افزایش داده، و برای چرخش به سمت چپ دور موتورهای چپ جلو و چپ عقب را زیاد می‌کنیم. البته لازم به ذکر است که برای کنترل مناسب ربات، می‌بایست کنترل روی هر چرخ دقیق صورت پذیرد که از لغزش اضافی و ناخواسته جلوگیری شود.

موتور

شکل 5 ربات چهار چرخ با ۴ موتور مجزا

دسته‌ی دوم به صورتی‌ست که چرخ‌های کناری به یکدیگر وصل هستند هر دو چرخ به یک موتور متصل اند. به این سیستم انتقال قدرت و پیکربندی موتورها، سیستم کنترل تانکی می‌گویند. بنابراین همانطور که در شکل 6 مشخص است، در این سیستم حرکتی تنها دو موتور وجود دارد. موتور سمت راست و موتور سمت چپ. برای حرکت دو چرخ سمت راست موتور سمت راست، و برای حرکت دو چرخ سمت چپ موتور سمت راست را درایو میکنیم. همچنین برای گردش به راست دور موتور سمت چپ را زیاد کرده، و برای گردش به چپ دور موتور سمت راست را زیاد می‌کنیم.

سیستم حرکتی تانکی برای چرخش درجا با سرعت بالا بسیار مناسب است. همچنین برای کنترل چرخ‌ها، نسبت به حالت پیشین،‌ لغزش ناخواسته‌ی کمتری را در عمل خواهد داشت.

دو چرخ پشت هم را به وسیله‌ی تسمه و یا گیربکس و... به یکدیگر می‌توان متصل کرد. از دلایل استفاده از این سیستم و متصل کردن چرخ‌های کناری به هم نیز به ‌این می‌توان اشاره نمود که در هنگام بالاروى از صخره­ها و موانع، بالارفتن يك چرخ موجب بالا رفتن چرخ ديگر مى­شود. به همين دليل در طراحى بدنه بايد موارد خاصی را در نظر گرفت.

موتور

شکل 6 ربات چهار چرخ با سیستم حرکتی تانکی

دسته سوم به اين صورت است كه دو چرخ عقب كار انتقال قدرت و درایو را انجام مى­دهند و دو چرخ جلو به صورت هرزگرد به اين طرف و آن طرف كشيده مى شوند. در حقیقت، مانند سیستم حرکتی خودرو، دو چرخ جلو نقش جهت دهی را دارند. در این حالت برای جهت دهی می‌توان از اتصال موتورهایی مانند سروو موتور به دو چرخ جلویی استفاده کرد.

موتور

شکل 7 ربات چهار چرخ با سیستم حرکتی دو موتور و فرمان

## مرور ادبیات

### کنترل همکارانه‌ی سیستم چند رباته

سیستم چند رباته[[9]](#footnote-9)،‌ سیستمی است که در آن تمام ربات‌ها با تبادل اطلاعات با یکدیگر همکاری می‌کنند تا وظایف کنترلی مختلفی را به انجام برسانند، چشم اندازهای کاربردی گسترده‌ای دارد. به دلیل کارآیی بالاتر، انعطاف بیشتر و تطابق با محیط‌های ناشناخته، و قابلیت همکاری، سیستم چند رباته می‌تواند به طور مناسب از منابع استفاده کند، قابلیت اتکا به سیستم را افزایش دهد، و کیفیت اجرای وظایف را نسبت به یک سیستم تک ربات بهبود بخشد. در یک سیستم همکارانه چند رباته‌ي متحرک، وظیفه و پیچیدگی تک تک ربات‌ها متمرکز، و تقویت می‌شود. بنابراین، همکاری بین ربات‌هاست که توانایی سیستم را بیان می‌کند. این مزایای مرتبط با سیستم‌های چند رباته مبتنی بر همکاری بین ربات‌ها، توجه محققین را بسیار افزایش داده و در سال‌های اخیر تلاش‌های زیادی برای تولید روش‌های کنترل همکارانه چند رباتی شده است. [1-6].

در بحث تعقیب مسیر زمانی[[10]](#footnote-10)، در کنترل سیستم چند رباته‌ی متحرک، پیشرفت‌های بسیاری حاصل شده است [7-16]. به ویژه، نویسندگان مقاله [7] چارچوب[[11]](#footnote-11) جدیدی برای مدل‌سازی فرم‌دهی[[12]](#footnote-12) ربات‌های متحرک بر اساس نظریه گراف ایجاد کردند و تغییر فرم‌دهی را به تغییر ساختار گراف مرتبط کردند. در [9-11]، کنترل فرم‌دهی چند ربات متحرک غیرهولونومیک، به یک مسئله‌ی وفاق[[13]](#footnote-13) در حالت‌های سیستم تبدیل شده و سپس کنترلر سینماتیک توزیع‌شده و کنترلر دینامیکی تطابقی طراحی شده است، تا تمام ربات‌ها به سمت به مسیرهای زمانی مرجع حرکت کنند و به طور مجانبی[[14]](#footnote-14) به الگوی هندسی مطلوب همگرا شوند. با وجود تحقیقات متعدد، هنوز چالش‌های زیادی باقی مانده است، یکی از آن‌ها نامعلومی[[15]](#footnote-15) یا عدم قطعیت مدل غیرخطی است. عدم قطعیت‌های مدل‌سازی نقش بسیار مخربی در سیستم‌های کنترل توزیع‌شده غیرخطی[[16]](#footnote-16) دارند و باعث می‌شوند کنترل حرکتی دشوارتر شود. در [17] و [18]، فرض شده است که مدل سیستم معلوم است و عدم قطعیت غیرخطی مد نظر گرفته نشده است. در [19]، روشی برای طراحی کنترلر دینامیکی ارائه شده است که زمانی که دینامیک ربات معلوم است، و ورودی کنترلی گشتاور دنبال‌کننده، شامل دینامیک خود و دینامیک رهبر گروه[[17]](#footnote-17) است. در [20-22]، کنترل فرم‌دهی توزیع‌شده ربات متحرک چرخدار غیر هولونومیک با استفاده از شبکه عصبی انجام شده‌است. به طوریکه برای حل دینامیک بدون ساختار و مدل‌نشدهی سیستم ربات و تخمین آن از شبکه‌ی عصبی استفاده شده است.

سیستم‌های ربات‌های متحرک معمولاً وظایف نسبتاً ساده اما تکراری انجام می‌دهند. از دیدگاه توسعه فناوری، بهبود کارایی این سیستم‌ها و کاهش مصرف انرژی آن‌ها مبحثی مهم و غیرقابل چشم پوشی است. استفاده‌ی منطقی و مؤثر از دانش به دست آمده از فرآیند کنترل، می‌تواند از بسیاری از رفتارهای ناموثر جلوگیری کرده و وظایف کنترلی را به بهترین شکل اجرا کند، به عبارت دیگر، توسعه‌ی روش‌های کنترل مبتنی بر یادگیری می‌تواند در کاربردهای عملی مهندسی مزایای اجتماعی و اقتصادی بسیاری داشته باشد. در روش‌های کنترل مبتنی بر یادگیری ذکر شده، با وجود اینکه مساله‌ی وفاق بین همه‌ی ربات‌ها قابل پیاده سازی‌ست، اطلاعات وزن‌های شبکه عصبی در فرایند کنترل به طور کامل مورد بررسی قرار نمی‌گیرد. نظریه یادگیری تعیینی[[18]](#footnote-18) با استفاده از شبکه عصبی RBF به طور کامل مورد تحقیق واقع شده است. مانند [23-26]، جایی که دینامیک حلقه بسته‌ی ناشناخته‌ی سیستم غیرخطی، می‌تواند به دقت توسط شبکه عصبی RBF تقریب زده شود و شبکه‌ی آموزش دیده می‌تواند در کنترل یادگیری مجدد استفاده شود. با این حال، وقتی که نظریه‌ی یادگیری تعیینی در سیستم‌های چند رباته استفاده می‌شود، یادگیری شبکه‌های عصبی به طور کاملاً توزیع‌شده اتفاق می‌افتد، یعنی هر ربات وزن‌های شبکه‌ی عصبی خود را با دیگران به اشتراک نمی‌گذارد. در واقع هر ربات به صورت مستقل یادگیری تخمین سیستم غیرخطی را انجام می‌دهد. شبکه‌ی عصبی هر ربات تنها برای مسیر زمانی مرجع خاصی که به هر ربات اختصاص داده شده است، قابل استفاده است، که منجر به این می‌شود که تعمیم مدل شبکه عصبی آموزش دیده محدود شود. از [27] الهام گرفته شده است که مزیت توان یادگیری شبکه عصبی می‌تواند استفاده شود. زمانی که به ربات‌ها اجازه‌ی به اشتراک گذاشتن وزن‌های عص داده شود می‌توان این مزیت را دید. فضای تخمین توسط شبکه عصبی، در حالی می‌تواند گسترش یابد که وزن‌های تمام ربات‌ها به یک مقدار بهینه‌ی مشترک همگرا شوند.

### قیود حالت

به علاوه، با توجه به ایمنی در عمل، محدودیت‌های فیزیکی موتورها، امکان ساخت مکانیکی و سایر عوامل، معمولاً در سیستم کنترلی واقعی محدودیت‌های[[19]](#footnote-19) مختلفی ایجاد می‌شود. با توجه ملاحظات عملکرد و ایمنی سیستم ربات یاد شده، به محدود کردن حوزه عملیاتی برخی متغیرهای کلیدی (مانند متغیرهای خروجی سیستم)، قیود حالت[[20]](#footnote-20) گفته می‌شود. نادیده گرفتن قیود حالت ممکن است به تخریب سیستم به طور جدی، ناپایداری سیستم و حتی خرابی تجهیزات منجر شود. برای ارايه‌ی کنترلر با در نظر گرفتن قیود حالت در سیستم‌های غیرخطی، برخی از روش‌های مناسب پیشنهاد شده‌اند، مانند کنترل پیش‌بینی مدل[[21]](#footnote-21) [28، 29]، تابع تبدیل خطا[[22]](#footnote-22) [30] و تابع لیاپانوف مانع[[23]](#footnote-23) (BLF)[31]. برای سیستم‌های غیرخطی فیدبک دار با قیود خروجی، یک روش کنترل با استفاده از تابع تانژانت محور tan-BLF در [32] پیشنهاد شده است تا دنبال کردن بهینه برای مسیر زمانی مرجع بدون نقض قیود را تضمین کند. در [33]، یک کنترل کننده شبکه عصبی تطابقی با استفاده از روش تابع BLF برای حل مساله‌ی کنترلی دنبال کردن مسیر ربات n-link، با در نظر گرفتن قیود تمامی حالت‌های سیستم عدم قطعیت در مدلسازی ارائه شده است. در کنترل همکارانه‌ی چند ربات، اگر دنبال کردن مسیر به طور پایدار و با رعایت قیود حالت هر تک ربات ممکن شود، عملکرد کنترلی سیستم ربات‌ها می‌تواند به طور چشمگیری بهبود یابد. بدین ترتیب از برخورد ربات‌ها با یکدیگر و این قبیل حوادث می‌توان جلوگیری نمود.

### خلاصه‌ی فصول

با توجه به بحث‌های فوق، در این تحقیق به ارایه‌ی کنترلری همکارانه مبتنی بر یادگیری دینامیک غیرخطی، برای تعقیب مسیر زمانی چند ربات چرخدار یکسان با در نظر گرفتن قیود حالت پرداخته شده است. در این تحقیق، تمام ربات‌ها در سیستم چند رباته به عنوان سیستم‌های غیر قطعی[[24]](#footnote-24) یکسان در نظر گرفته می‌شوند و هر ربات مسیر زمانی مرجع مختص به خود را دارد. با استفاده از شبکه‌ی عصبی با تابع فعالیتRBF[[25]](#footnote-25) برای تخمین دینامیک غیرخطی ناشناخته، یک کنترلر دینامیکی همکارانه ارائه خواهد شد.

در فصل دوم به مدلسازی سینماتیکی ربات، ارائه‌ی کنترلر سینماتیکی به همراه توضیح مفهوم تابع لیاپانوف مانع و شبیه سازی مرتبط با اعمال کنترلر به شرایط سینماتیکی ربات می‌پردازیم.

سپس در فصل سوم مبتنی بر تابع لیاپانوف مانع، با استفاده از نظریه‌ی وفاق، و توضیح توپولوژی شبکه‌ی ارتباطی بین ربات‌ها، کنترلر دینامیکی را ارائه نموده و تخمینگر شبکه عصبی RBF برای تخمین دینامیک ربات را معرفی می‌کنیم. سپس نتایج شبیه سازی حاصل از اعمال کنترلر به دینامیک ربات را ارائه میکنیم.

در فصل چهارم به طراحی مکانیکی و الکترونیکی ۳ پلتفرم ربات چرخدار، و چگونگی ایجاد فیدبک مکان بر پایه‌ی پردازش تصویر و پیکربندی رنگی هر پلتفرم توضیح داده خواهد شد.

و در فصل پنجم نتایج حاصل شده از پیاده سازی الگوریتم بر روی ۳ پلتفرم ربات چرخدار در حرکت واقعی آن‌ها ارائه می‌شود.

# فصل دوم

**طراحی کنترلر سینماتیکی**

## مقدمه

در این بخش از پایان نامه، ابتدا مفهوم تابع لیاپانوف و چگونگی اثبات پایداری و سپس تابع لیاپانوف مانع توضیح داده می‌شود. این توابع برای رعایت قیود حالت بسیار پرکاربرد و ملزوم هستند. در ادامه برای ارائه‌ی کنترلر سینماتیکی و همچنین دینامیکی، از این توابع استفاده خواهیم نمود. سپس برای یک پلتفرم ربات چرخدار دارای سه چرخ، مدلسازی سینماتیکی را ارائه می‌کنیم. این مدلسازی به طور کامل سرعت خطی و دورانی ربات را با مکان مورد نظر در صفحه‌ی مختصات دکارتی، پیوند می‌دهد. سپس برای این مدل یک کنترلر سینماتیکی ارائه نموده و با استفاده از شبیه سازی و تابع لیاپانوف مانع،‌ پایداری سیستم را اثبات نموده و در نهایت نتایج شبیه سازی ارائه می‌شوند.

## تابع لیاپانوف

در نظریه‌ی معادلات دیفرانسیل عادی یا ODE، توابع لیاپانوف، توابعی اسکالر هستند که می‌توان از آن‌ها برای اثبات پایداری نقطه‌ی تعادل[[26]](#footnote-26) ODE استفاده نمود. این معادلات ODE می‌توانند همان معادلات فضای حالت سیستم و بیان کننده‌ی رفتار سیستم باشند. این توابع به نام که به نام الکساندر لیاپانوف[[27]](#footnote-27) شناخته می‌شوند. استفاده از توابع لیاپانوف شامل دو روش اصلی است، که روش دوم برای اثبات پایداری مرسوم و جامع تر است. توابع لیاپانوف در تئوری پایداری سیستم‌های دینامیکی و تئوری کنترل اهمیت بسیاری دارند. چرا که اثبات پایداری مبحث مهمی در کنترل به شمار می‌رود. همچنین یک مفهوم مشابه در نظریه‌ی زنجیره‌های مارکوف[[28]](#footnote-28) برای فضای حالت[[29]](#footnote-29) عمومی نیز وجود دارد، که معمولاً با نام توابع فاستر-لیاپانوف شناخته می‌شود.

برای انواع خاصی از ODEها، یا همان معادلات فضای حالت سیستم‌ها، وجود توابع لیاپانوف شرط لازم و کافی برای پایداری است. در حالی که هیچ روش عمومی برای ساخت توابع لیاپانوف برای معادلات سیستم وجود ندارد، در بسیاری از موارد خاص، ساخت توابع لیاپانوف دشوار نیست. به عنوان مثال، برای سیستم‌های با یک حالت، انتخاب توابع چند جمله‌ای کافی ست. همچنین حل یک نامساوی از یک ماتریس خطی خاص، تابع لیاپانوف را برای سیستم‌های خطی می‌سازد، و در آخر استفاده از قوانین حفظ[[30]](#footnote-30) معمولاً می‌تواند برای ساخت توابع لیاپانوف برای سیستم‌های فیزیکی استفاده شود.

اگر سیستم ما به صورت زیر باشد

آنگاه اگر نقطه‌ی x = 0، نقطه‌ی تعادل سیستم باشد تابع لیاپانوف V(x) به صورتی تعریف می‌شود که پیوسته،‌ دارای مشتق مرتبه اول، و برای x ≠ 0 همیشه مثبت است. همچنین شرط زیر به عنوان تابع نامحدود شعاعی مطرح است.

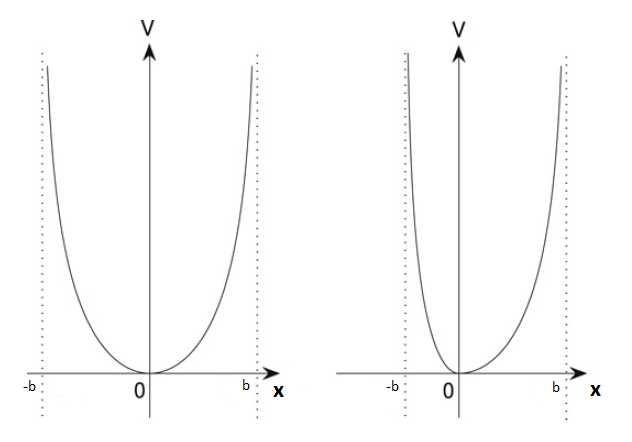
حال چگونگی استفاده از تابع کاندید لیاپانوف برای اثبات پایداری به این صورت است که اگر تابع کاندید لیاپانوف با شرایط بالا انتخاب شد، حال می‌بایست تابع مشتق بر حسب زمان آن بدست آید. اگر این تابع مشتق زمانی، به ازای هر x در محدوده‌ی تعریف x،‌ منفی معین باشد، سیستم ما در نقطه‌ی تعادل پایدار مجانبی گلوبال یا پایدار لیاپانوف تعریف می‌شود.

## تابع لیاپانوف مانع

همانطور که اشاره شد، مبحث رعایت و در نظر گیری قیود برای حالت‌ها، ورودی‌ها، و خروجی کنترلی، بسیار حائز اهمیت است. چرا که در مواردی سلامت عملکردی سیستم ربات، در عمل، مشروط و منوط به تخطی نکردن از این قیود و رعایت آن‌هاست. برای نیل به این هدف و رعایت این قیود، در ارائه‌ی کنترلرها از توابع لیاپانوف مانع یا به اختصار BLFها استفاده می‌شود. این توابع اولین بار در [32] مطرح شد و پس از آن در مقالات بسیاری به آن ارجاع داده شد. بنابراین این توابع نقش مهمی را در تولید کنترلرهای مبتنی بر رعایت قیود ایفا می‌کنند.

برای تعریف این توابع باید گفت یک تابع لیاپانوف مانع یک تابع اسکالر V(x) است که با توجه به سیستم Pₓ = ƒ(x) در یک منطقه باز D که مبتنی بر x است، تعریف می‌شود. این تابع پیوسته[[31]](#footnote-31)، مثبت معین، دارای مشتقات جزئی مرتبه اول در هر نقطه از D است. خاصیت بسیار مهم این تابع این است که هر گاه x به عنوان ورودی تابع،‌ به مرز D نزدیک می‌شود،V(x) به ∞ میل می‌کند. همچنین به ازای یک مقدار مثبت ثابت b شرط-b ≤ V(x(t)) ≤ b برای همه t ≥ 0در xهای حاصل از حل برقرار است. همچنین می‌بایست شرط x(0) ∈ D برقرار باشد و مقدار ورودی در لحظه‌ی 0 عضوی از بازه‌ی D باشد.

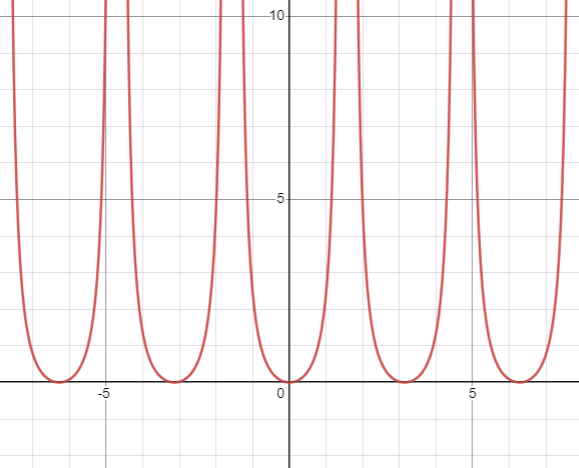
همانطور که در شکل زیر نشان داده شده است، تابع لیاپانوف مانع ممکن است به صورت متقارن یا یا نامتقارن باشد.



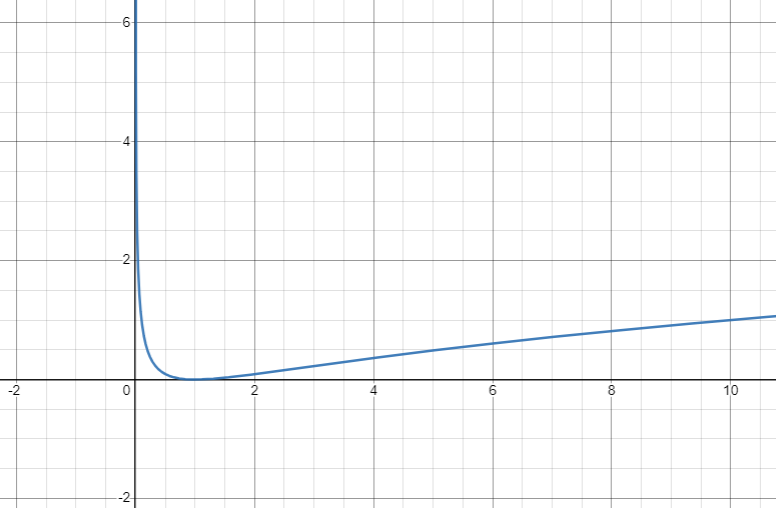
شکل 8 تابع لیاپانوف مانع. سمت راست نامتقارن. سمت چپ متقارن

با توجه به آنچه گفته شد، از خصوصیات مهم توابع لیاپانوف مانع آن است که این توابع دارای دامنه‌ی به خصوصی و برد آن‌ها شامل مثبت بی‌نهایت و همیشه مثبت باشد. چرا که می‌بایست با نزدیک شدن متغیر ورودی به مرزهای مشخصی، مقدار خروجی تابع به بی نهایت میل کند. همچنین می‌دانیم انتخاب تابع کاندید لیاپانوف بسته به نوع معادلات سیستم می‌تواند متفاوت باشد. از همین روی انواع مختلفی از توابع لیاپانوف مانع قابل انتخاب هستند.

از جمله‌ی این توابع می‌توان به مربع تانژانت، مربع لگاریتم و لگاریتم بر پایه‌ی نپر و... اشاره کرد. این توابع به ازای ورودی‌های خاصی به بینهایت میل می‌کنند.



شکل 9 تابع مربع تانژانت که به ازای به بینهایت میل می‌کند



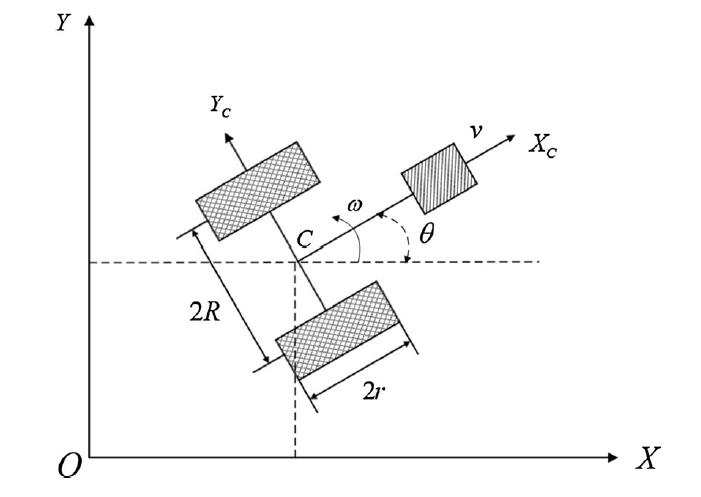
شکل 10 تابع مربع لگاریتم که به ازای x = 0 به بینهایت میل می‌کند

بعنوان مثال تابع زیر می‌تواند یک تابع کاندید لیاپانوف مانع باشد.

مشخصاً در تابع فوق اگر مقدار x به 5 میل کند، مقدار V(x) به بینهایت میل می‌کند. همچنین این تابع مثبت معین است.

## مدل سازی سینماتیکی

برای ارائه‌ی کنترلر سینماتیکی ابتدا می‌بایست پلتفرم را در فضای سینماتیک مدل سازی کنیم. برای این مقصود یک پلتفرم را به شکل زیر در صفحه‌ی مختصات دکارتی در نظر می‌گیریم.



شکل 11 نمونه‌ی یک پلتفرم ربات چرخ دار غیر هولونومیک با سه چرخ

در این شکل نمونه‌ی یک پلتفرم ربات چرخ دار غیر هولونومیک با سه چرخ مشاهده می‌شود. نقطه‌ی C مرکز ثقل ربات که در نقطه‌ی ( واقع است در نظر گرفته می‌شود. دو چرخ عقب متصل به موتور و برای انتقال قدرت به سیستم و چرخ جلو هرزگرد می‌باشد. مقدار r شعاع چرخ و 2R برابر با عرض ربات می‌باشد. مقدار برابر با زاویه‌ی چرخش ربات و در جهت پادساعتگرد مثبت است. بنابراین در این جهت سرعت زاویه‌ای مثبت خواهد بود. همچنین سرعت خطی v در جهت بردار است.

بنابراین برای مدلسازی سینماتیکی، معادله‌ی ماتریسی زیر مطرح می‌شود

به این صورت معادلات دیفرانسیل حاکم بر سیستم مشخص شده و ارتباط دهنده‌ی مکان ربات و سرعت خطی و دورانی آن تبیین می‌گردد.

در اینجا برای تسهیل ادامه‌ی روند و ارائه‌ي کنترلر سینماتیکی، از یک تبدیل همومورفیسم[[32]](#footnote-32) برای تبدیل المان‌های مکانی سیستم، و یک ماتریس تبدیل حالت برای تبدیل سرعت‌های خطی و دورانی سیستم بهره می‌گیریم.

در این صورت سیستم می‌تواند به حالت معادلات زنجیری زیر در بیاید

## کنترلر سینماتیکی

حال برای ارائه‌ي کنترلر سینماتیکی،‌ ابتدا باید خطاهای حرکتی را تعریف کنیم. اگر مقدار دلخواه را با اندیس d نشان دهیم،‌ خطاهای سرعت تبدیل شده به صورت زیر تعریف می‌شوند

که در آن یک متغیر واسط و شامل می‌باشد که تعریف آن ارائه خواهد شد.

حال به سراغ تعریف مقادیر قیود متغیرهای حالت بر روی و و می‌رویم. همانطور که مشخص است از آنجا که و قیدی بر روی معادلات نخواهیم داشت، پس قیود روی دو حالت دیگر تعریف می‌شوند. حال معادلات زیر را در فرض می‌گیریم.

که در آنها و همان قیود متغیرهای حالت ما هستند. در حقیقت می‌خواهیم که متغیرهای حالت ما از این دو مقدار تجاوز نکنند. از طرفی برای مقادیر دلخواه متغیرهای حالت نیز قیودی در نظر می‌گیریم.

بنابراین، می‌توان از معادلات بالا به قیودی روی و رسید.

بنابراین، مشخص نمودیم که میخواهیم خطاهای و از مقادیر و تجاوز نکنند.

همچنین با توجه به معادلات زیر

و مشتق گیری از طرفین معادلات زیر را داریم

حال یک تابع کاندید لیاپانوف مانع به صورت زیر تعریف می‌کنیم.

در معادلات بالا همانطور که مشخص است، تابع مثبت معین بوده و اگر خطای یا به مقادیر و نزدیک شوند، تابع به بینهایت میل می‌کند.

همچنین این تابع پیوسته و دارای مشتق مرتبه اول است. بنابراین یک کاندید مناسب به عنوان تابع لیاپانوف مانع محسوب می‌شود. علاوه براین،‌ با توجه به یکسان بودن L پلتفرمی که کنترل همکارانه را بر روی آن‌ها اعمال خواهیم کرد، میتوان این معادلات را به شکل زیر برای L پلتفرم تعمیم داد.

با مشتق مرتبه‌ی اول نسبت به زمان از تابع لیاپانوف مانع بالا داریم:

بنابراین کافی‌ست مقادیر و را به گونه‌ای طراحی کنیم که منفی معین شده و پایداری سیستم با در نظر گیری قیود حالت اثبات شود. برای این منظور داریم

بنابراین و را طراحی نمودیم. حال با جایگذاری در داریم

در معادله‌ی بالا ترم سمت چپ یعنی یک عبارت همیشه منفی به ازای و است.

ترم سمت راست یعنی نیز به ازای طراحی مناسب، به سمت 0 خواهد رفت. بنابراین می‌بایست را به درستی طراحی کنیم. از معادله‌ي و با توجه به اینکه مقدار مشمول قید نمی‌شود، می‌توان به سادگی به مقدار زیر رسید.

در این صورت نیز، یک تابع لیاپانوف ساده به صورت زیر مطرح می‌شود.

برای منفی شدن باید باشد.

بنابراین تابه اینجا مقدار تابع لیاپانوف برابر با را دیدیم و مقدار همیشه منفی شد. پس کنترلر سینماتیکی معرفی شده، سیستم سینماتیکی ما را پایدار خواهد کرد. همچنین با توجه به تعریف تابع لیاپانوف مانع ، میدانیم که خطاهای و و به تبع آنها و و به تبع آن‌ها و از قیود خود تخطی نمی‌کنند.

## شبیه سازی سینماتیکی

در این بخش ابتدا مقادیر عددی مسیر زمانی مرجع مربوط به هر ربات، قیود حالت و ضرایب کنترلی و... را ارائه کرده سپس شبیه سازی و نتایج آن در فضای متلب را نشان خواهیم داد. برای این منظور و برای نشان دادن عمومیت کنترلر ارائه شده، ابتدا برای یک پلتفرم، دو مسیر زمانی متفاوت دایره و 8 را شبیه سازی کرده سپس برای هر سه پلتفرم مسیر زمانی بیضی ارائه می‌شود. این نتایج نشان دهنده‌ی پایداری سیستم و کاهش خطای حالت‌ها به سمت 0 می‌باشند که تضمین کننده‌ی صحت کنترلر سینماتیکی ارائه شده هستند.

مقادیر عددی شبیه سازی، فارغ از شکل مسیر زمانی مرجع که شامل قیود حالت، و ضرایب کنترلی می‌باشند به صورت زیر هستند.

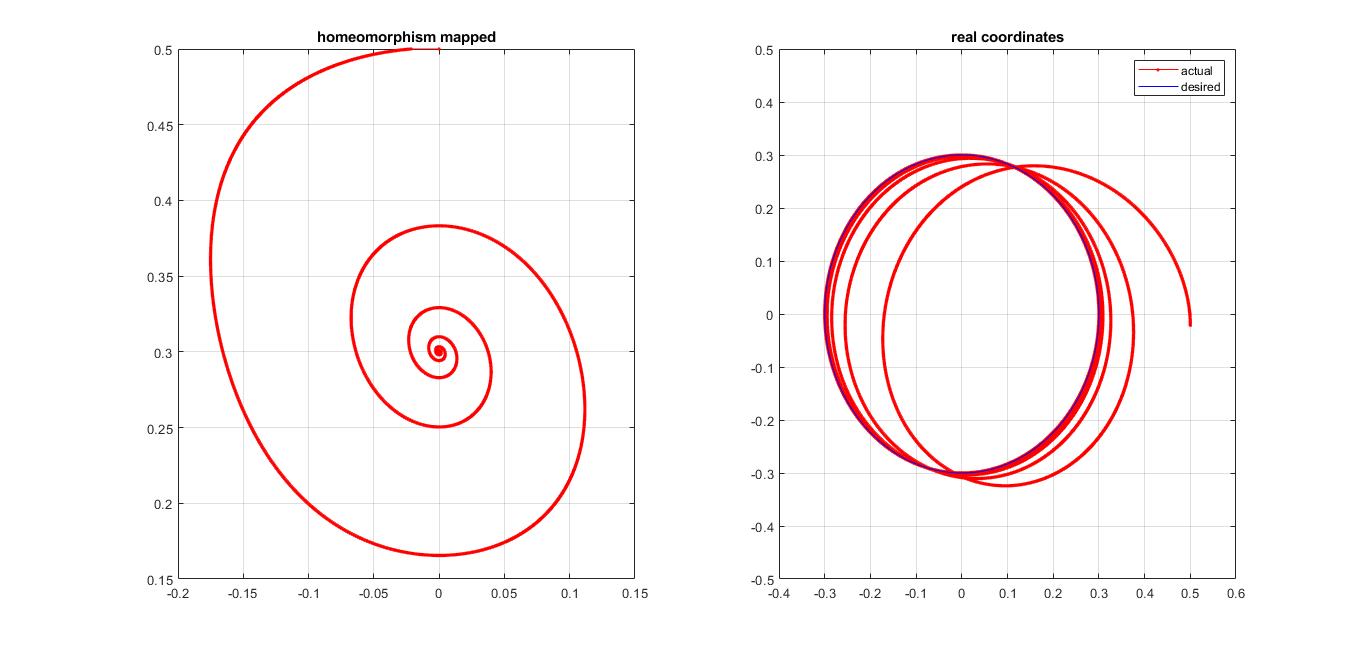
### دایره‌ی تک پلتفرم

#### مقادیر عددی

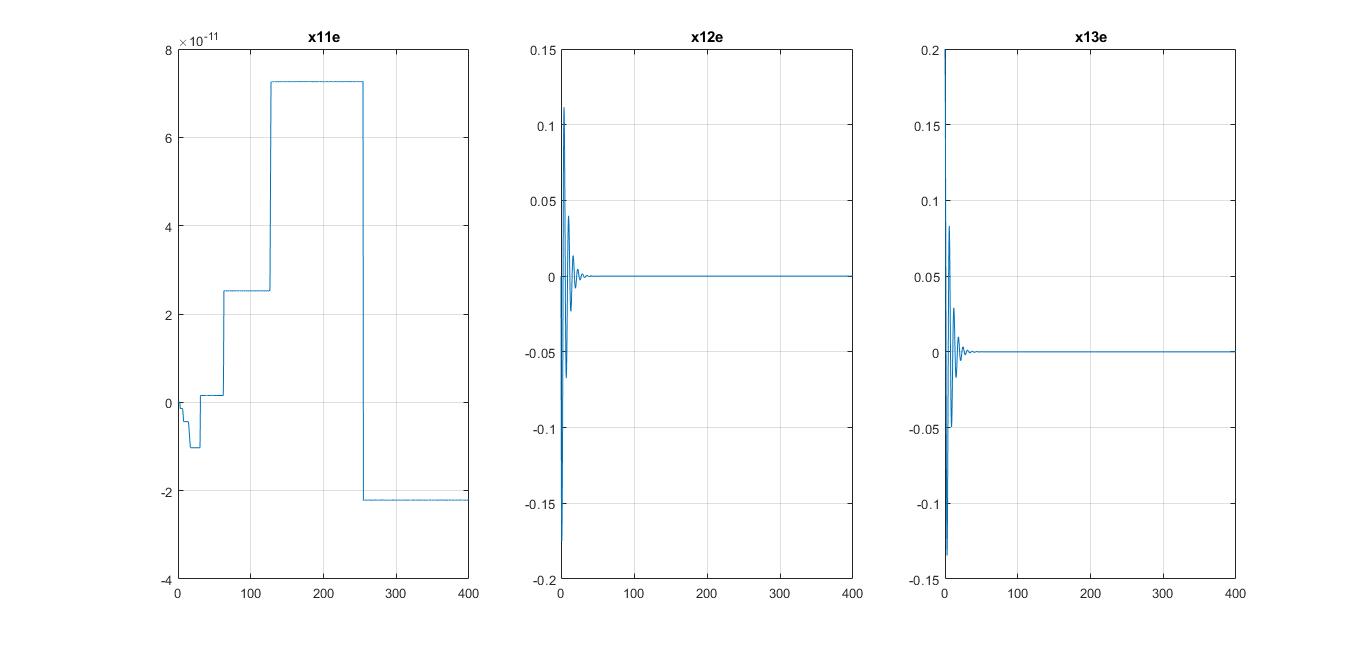
ابتدا برای تک پلتفرم یک مسیر ساده‌ی دایره در نظر می‌گیریم.

شرایط اولیه:

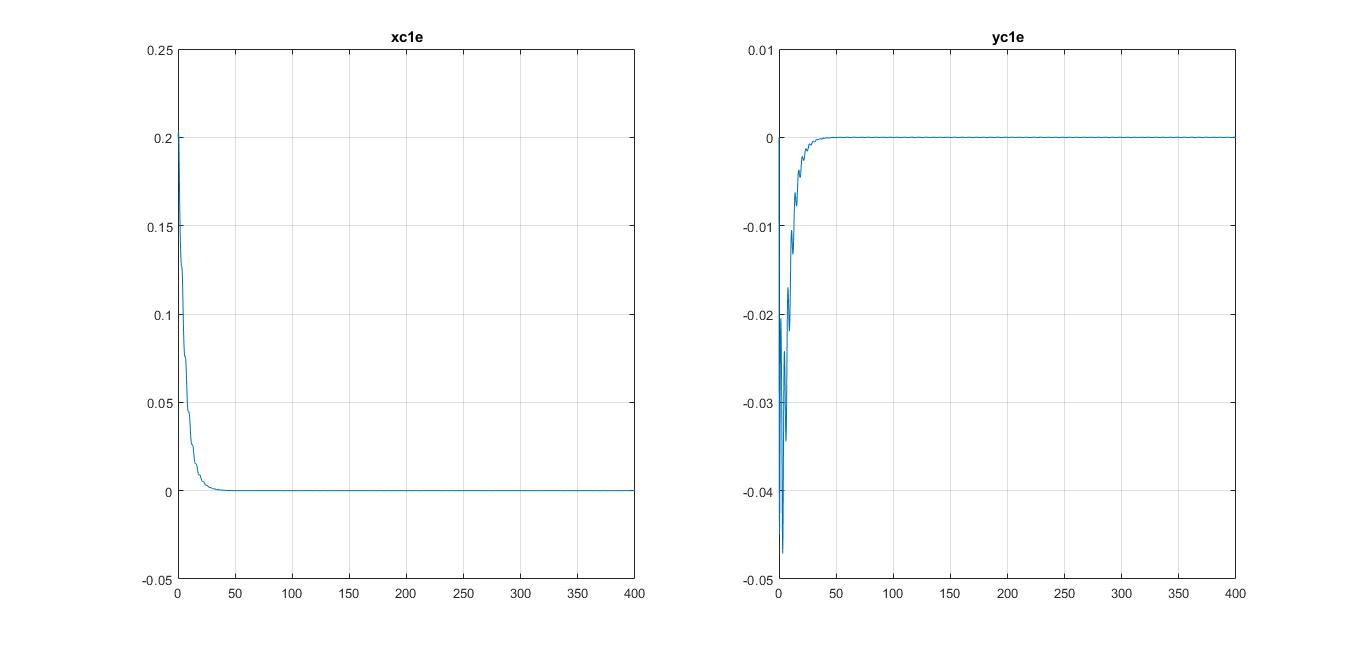
#### نتایج شبیه سازی



شکل 12 مسیر حرکت ربات سمت راست مختصات واقعی سمت چپ مختصات تحت تبدیل همومورفیسم



شکل 13خطای حالت‌های تبدیل شده



شکل 14 خطای حالت‌ها در مختصات واقعی

### 8 انگلیسی

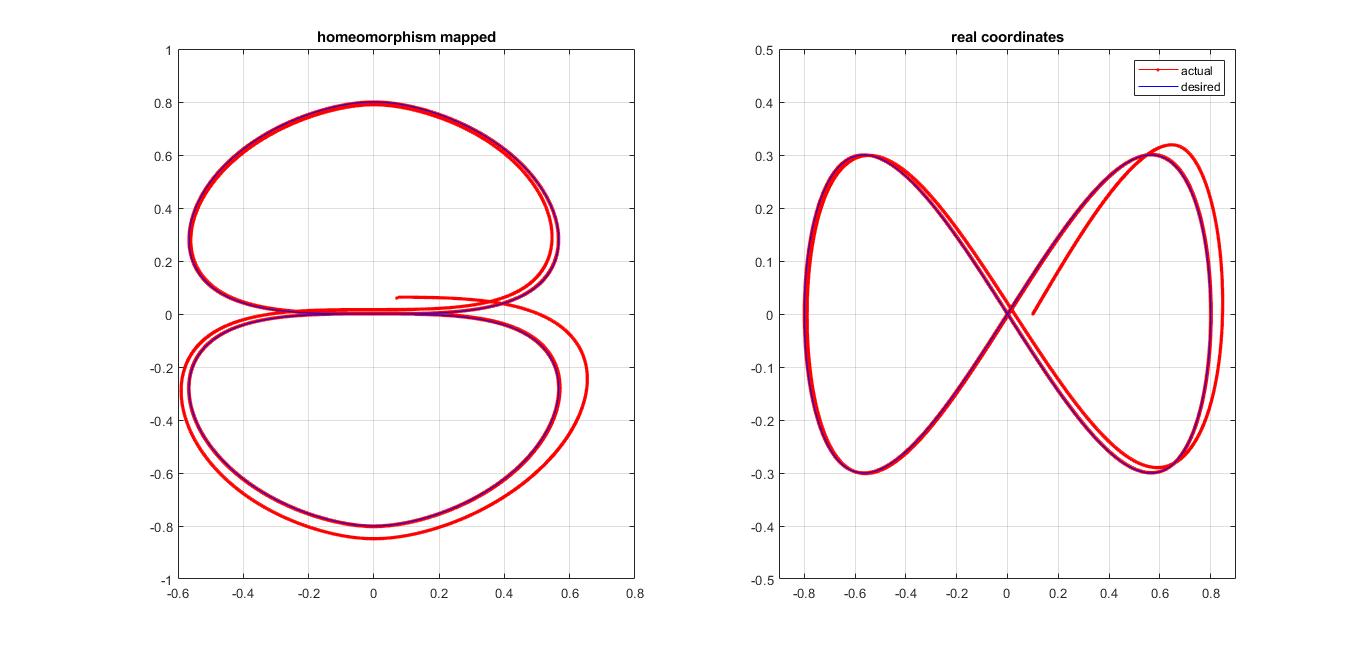
#### مقادیر عددی

ابتدا برای تک پلتفرم یک مسیر ساده‌ی دایره در نظر می‌گیریم.

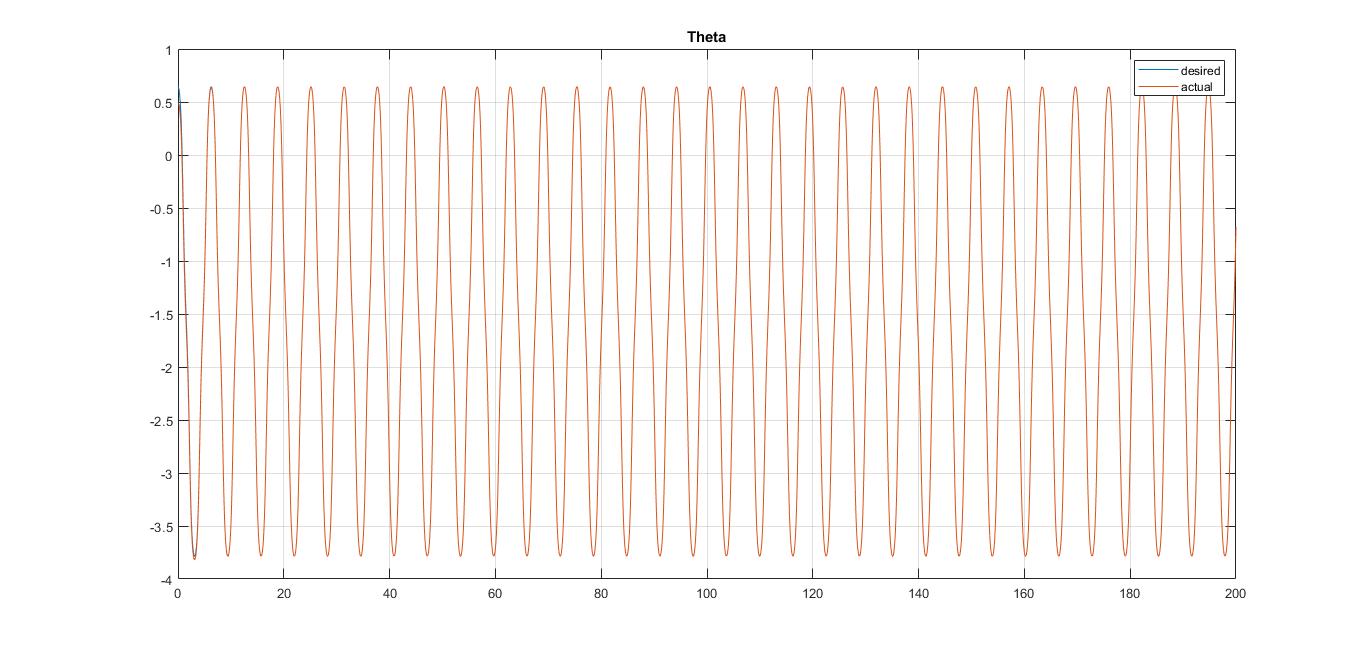
شرایط اولیه:

لازم به ذکر است که این مسیر زمانی از مبدا شروع شده و به سمت راست بالا می‌رود.

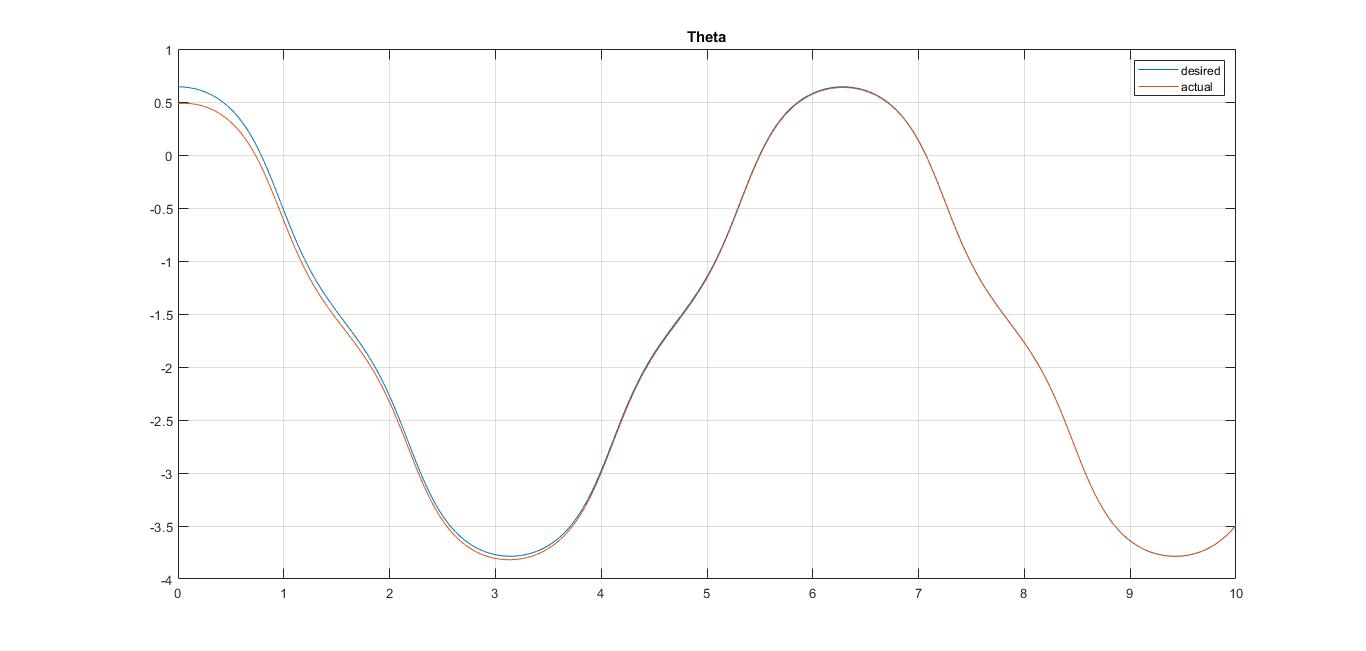
#### نتایج شبیه سازی



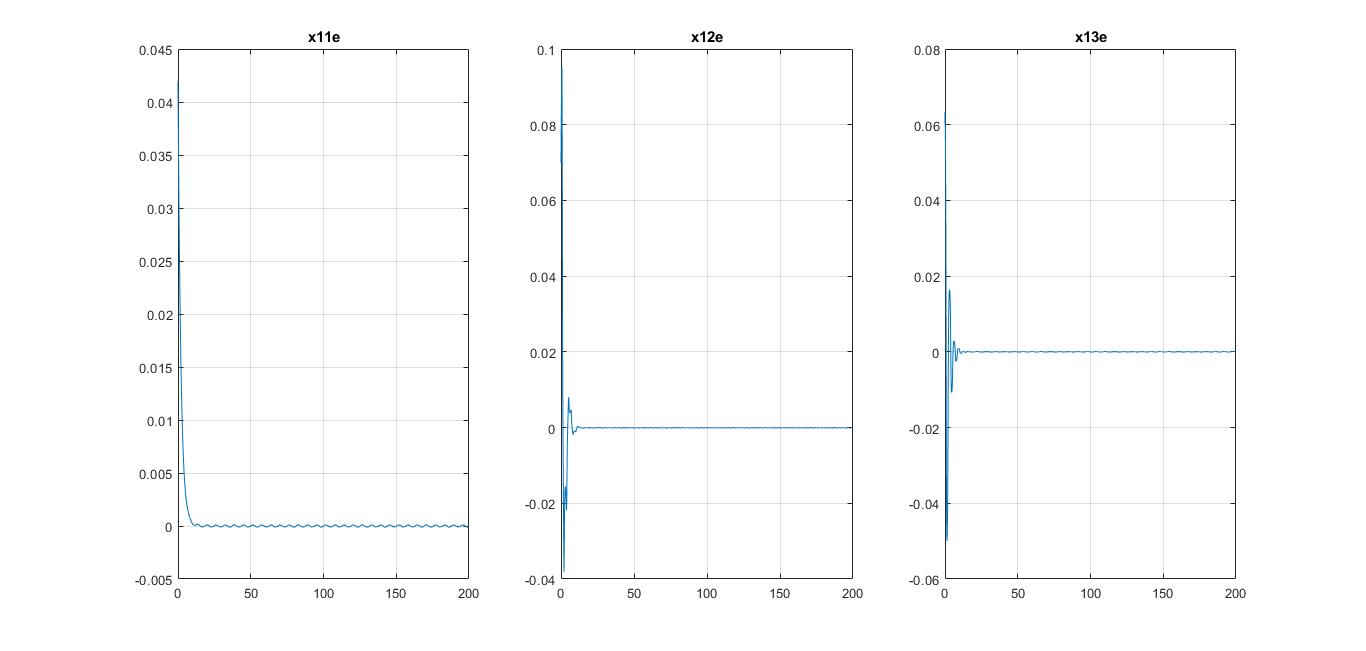
شکل 15 مسیر حرکت ربات سمت راست مختصات واقعی سمت چپ مختصات تحت تبدیل همومورفیسم



شکل 16 تغییرات زاویه‌ی دلخواه و واقعی



شکل 17 تغییرات زاویه دلخواه و واقعی تا زمان 10 ثانیه



شکل 18 خطای حالت‌های تبدیل شده



شکل 19 خطای حالت‌ها در مختصات واقعی

### بیضی برای سه ربات

#### مقادیر عددی

شبیه سازی‌های انجام شده برای تعداد L = 3 پلتفرم ارائه می‌شوند. همانطور که اشاره شد، به هر یک از ربات‌ها یک مسیر زمانی مجزا و متفاوت تخصیص داده می‌شود. همچنین می‌بایست و برای هر ربات با توجه به مسیر زمانی مرجع، محاسبه شود. به این صورت و همچنین مقدار است که برای هر ربات این مقادیر متفاوت خواهند بود. همچنین شرایط اولیه‌ی تخصیص داده شده به هر ربات متفاوت است. بنابراین مقادیر عددی به شرح زیر ارائه می‌شوند.

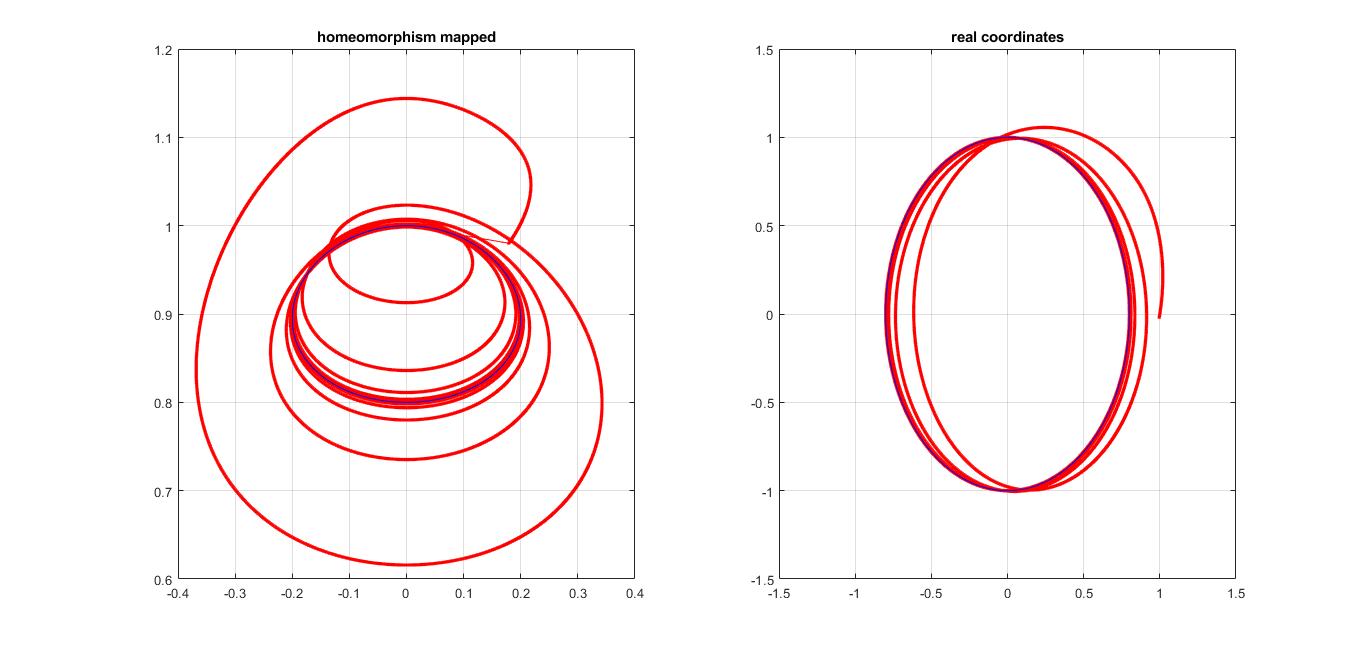
ربات 1:

ربات 2:

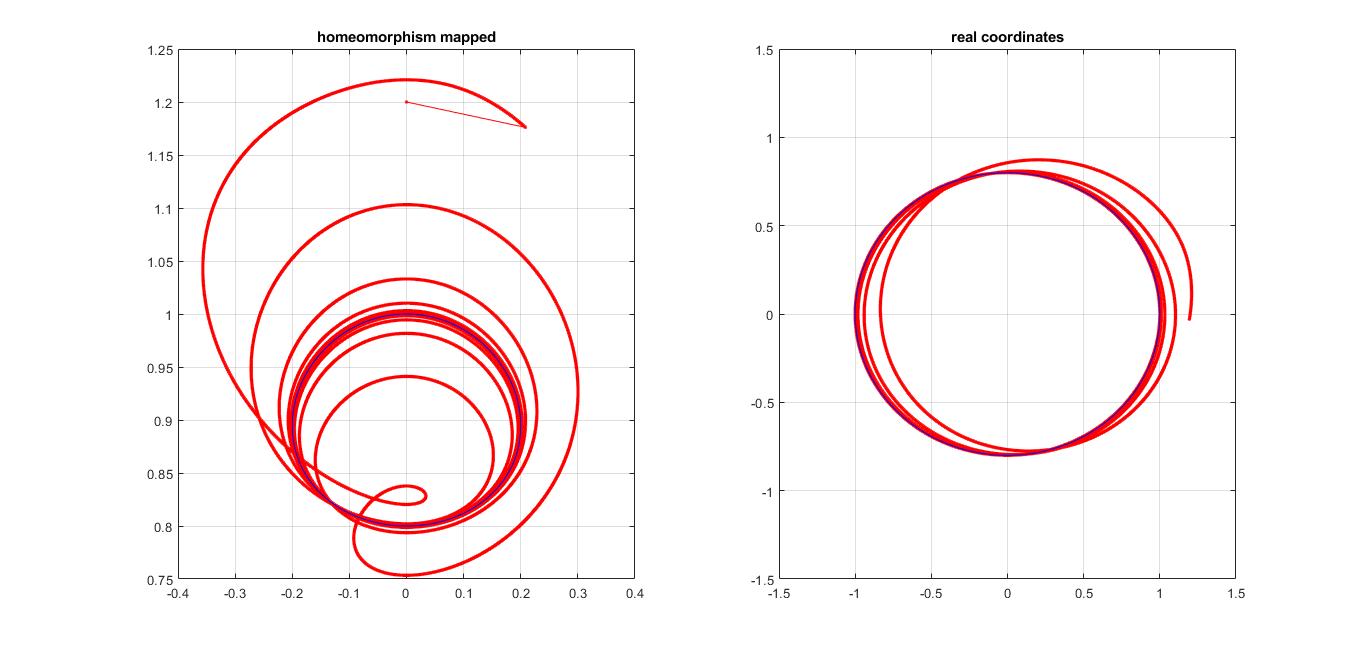
ربات 3:

شرایط اولیه:

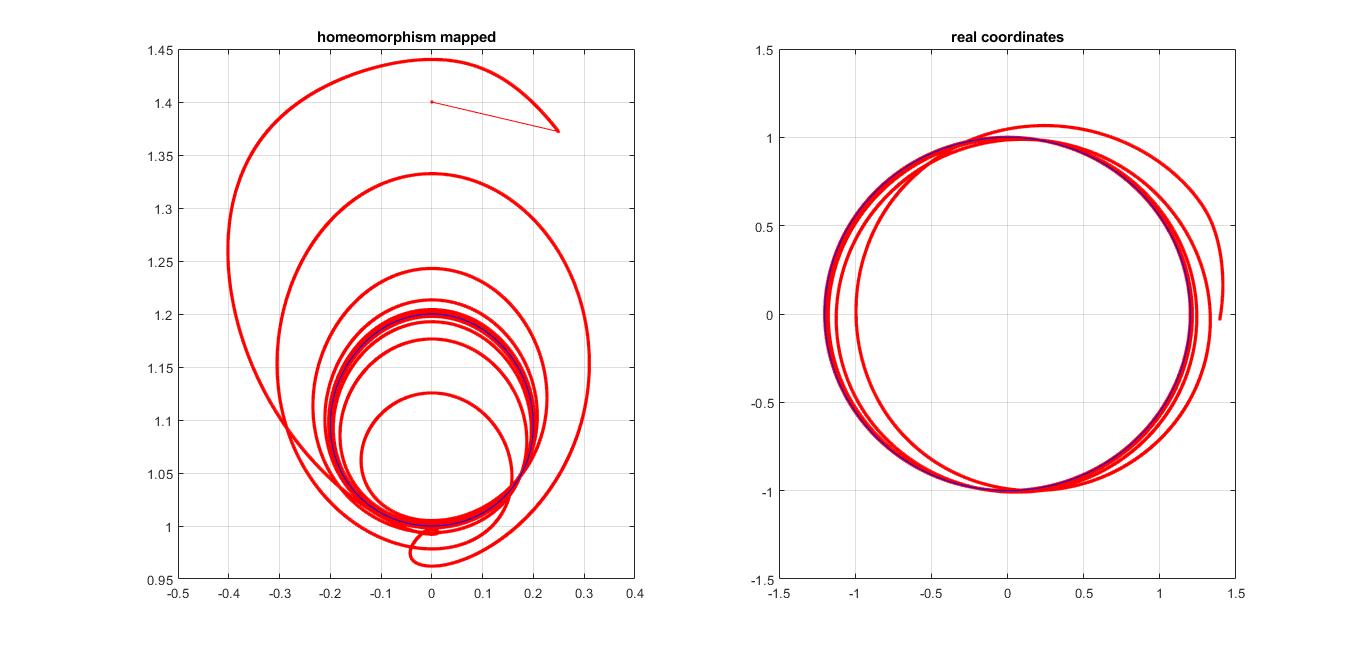
#### نتایج شبیه سازی



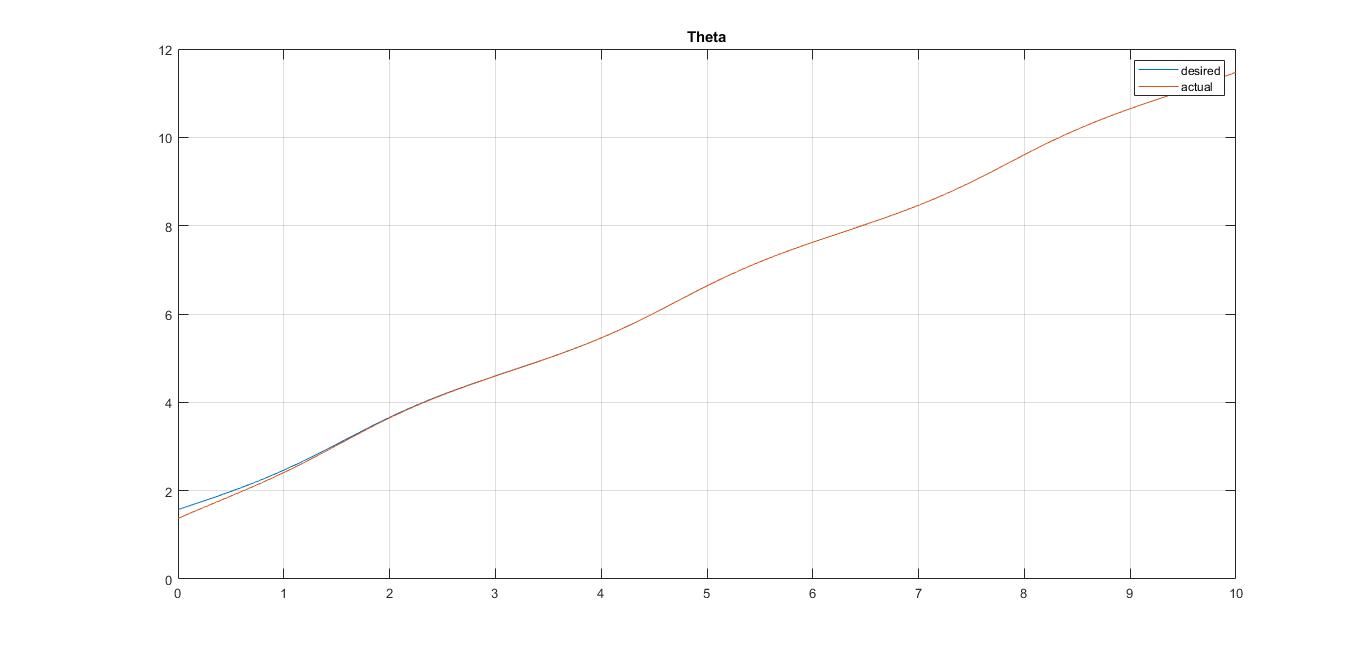
شکل 20 مسیر حرکت ربات 1 سمت راست مختصات واقعی سمت چپ مختصات تحت تبدیل همومورفیسم



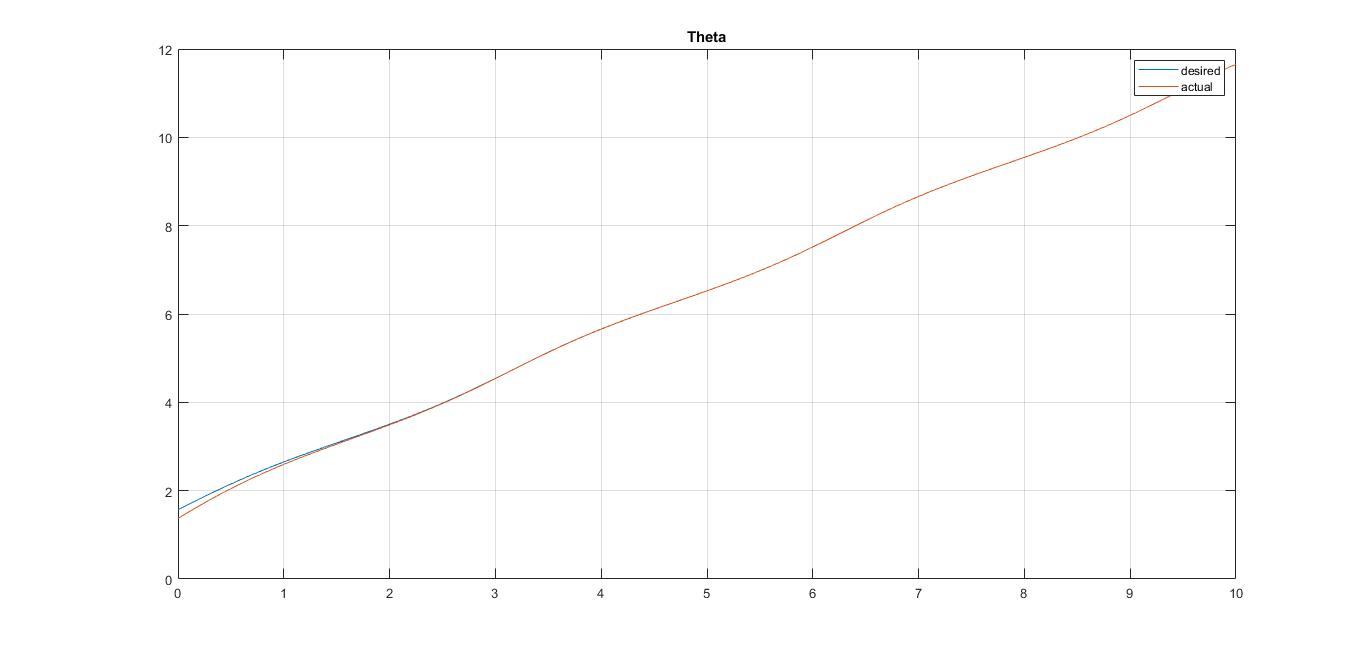
شکل 21 مسیر حرکت ربات 2 سمت راست مختصات واقعی سمت چپ مختصات تحت تبدیل همومورفیسم



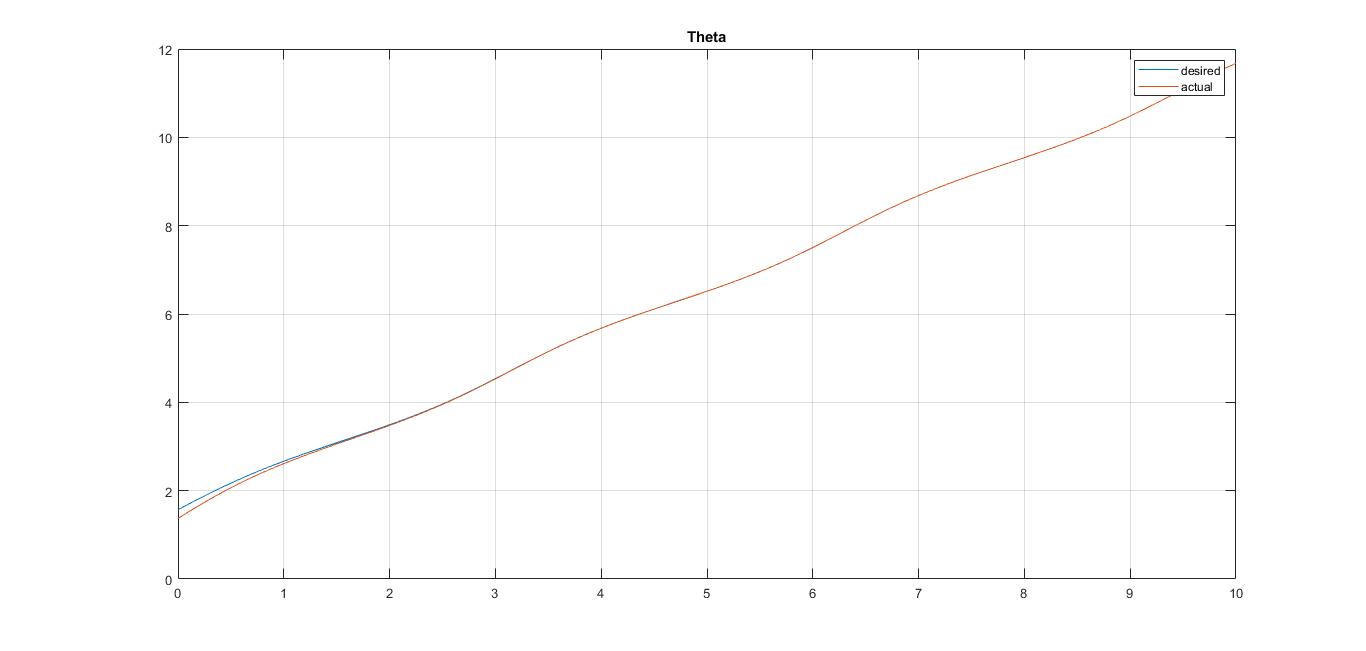
شکل 22 مسیر حرکت ربات 3 سمت راست مختصات واقعی سمت چپ مختصات تحت تبدیل همومورفیسم



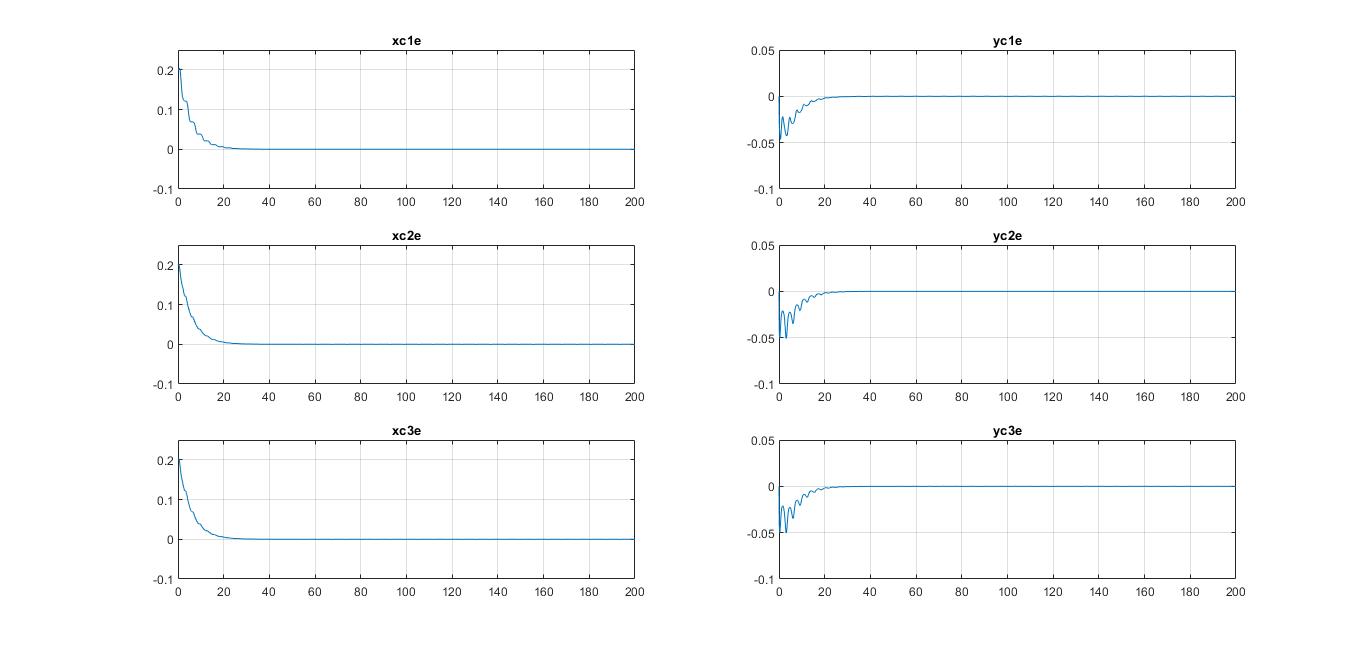
شکل 23 تغییرات زاویه دلخواه و واقعی ربات 1 تا زمان 10 ثانیه



شکل 24 تغییرات زاویه دلخواه و واقعی ربات 2 تا زمان 10 ثانیه



شکل 25 تغییرات زاویه دلخواه و واقعی ربات 3 تا زمان 10 ثانیه



شکل 26 خطای حالت‌های هر سه ربات در مختصات واقعی

## نتیجه گیری

با توجه به نتایج ارائه شده، می‌توان این گونه نتیجه گیری کرد که کنترلر سینماتیکی ارائه شده، موجب پایداری سیستم شده و خطای ناشی از شرایط اولیه‌ی مختلف به صفر میل می‌کند. همچنین عمومیت این کنترلر با توجه به تغییر مسیر زمانی مرجع به اشکال مختلف، مناسب است و سیستم در مسیر زمانی‌های پیچیده تر از دایره و بیضی نیز عملکرد مناسبی از خود نشان می‌دهد.

# فصل سوم

طراحی کنترلر دینامیکی

## مقدمه

امروزه، رویکردهای مبتنی بر یادگیری عمیق که از معماری شبکه‌های عصبی پیچشی استفاده می‌کنند، بدون شک ابزار اصلی در حل مسائل طبقه‌بندی تصاویر با کمک رایانه است. اولین شبکه عصبی پیچشی در سال 1998 به نام Lenet توسط لِکون[[33]](#footnote-33) مطرح شد [61]. حدود 50 هزار پارامتر و وزن جهت آموزش دارد. پس از آن با گذشت سالها، در سال 2012 اولین شبکه معروف پیچشی به نام Alexnet معرفی شد [57]. این شبکه بر روی مجموعه‌داده‌ای شامل بیش از یک میلیون تصویر در 1000 طبقه متفاوت به نام Imagenet بالاترین دقت را در مسابقه آن کسب کرد. تصاویر موجود در این مجموعه‌داده شامل منظره‌هایی از طبیعت، حیوانات و غیره می‌باشد [85]. در سال‌های بعد، معماری‌های دیگر شبکه عصبی پیچشی مانند GoogleNet (Inception) [86]، VGG [87]، Efficientnet [88] و Resnet [89] نیز ارائه شدند. اخیرا شبکه‌های جدیدتری مانند Swin [90]، ViT [91]، Regnet [92] و غیره ارائه شده اند که دقت‌های بالایی بر روی مجموعه‌داده ‌Imagenet از خود نشان داده اند. تمام مدل‌های آموزش دیده شده در این تحقیق، وزن‌های اولیه خود را از شبکه‌های آموزش دیده شده بر روی این مجموعه‌داده گرفته اند.

## یادگیری ماشین

یادگیری ماشین حوزه‌ای از هوش مصنوعی است که هدف آن ایجاد الگوریتم‌ها و تکنیک‌های جدید برای پیش‌بینی بر اساس داده‌های موجود و آموزش دیده شده است [93]. یادگیری ماشین مدل‌های كلي را با استفاده از داده‌های آموزشي توليد مي‌كند تا اين مدل‌ها بتوانند وجود يا عدم وجود يک الگو در داده‌های آزمون را تشخيص دهند. به طور کلی، روش‌‌های یادگیری ماشین به چهار دسته اصلی تقسیم می‌شوند. این دسته‌بندی بر اساس نوع یادگیری الگوریتم می‌باشد. دسته‌ی اول، یادگیری با نظارت[[34]](#footnote-34) نام دارد که در مجموعه‌داده‌های آموزشی، هر داده باید حداقل یک برچسب وجود داشته باشد تا مدل بتواند ورود‌ی‌ها را به این برچسب‌‌ها ربط دهد. دسته‌ی دوم، یادگیری بدون نظارت[[35]](#footnote-35) است که مجموعه‌داده‌های آموزش مدل بدون برچسب می‌باشند و مدل باید شباهتی میان دسته‌های مختلف پیدا کند: مانند الگوریتم‌های خوشه‌بندی. دسته‌ی بعدی، یادگیری نیمه نظارت[[36]](#footnote-36) است که در این نوع یادگیری، داده‌های آموزشی عمدتا بدون برچسب بوده و تنها بخش کمی از داده‌ها دارای برچسب می‌باشند [94]. دسته‌ی آخر یادگیری، یادگیری تقویتی[[37]](#footnote-37) نام دارد که بر اساس سیستم پاداش و تنبیه مدل را آموزش می‌دهد، به طوری که اگر مدل کاری کند که در آینده به هدف نزدیک‌تر شود، به آن امتیاز داده و در صورت انجام عکس آن، از امتیاز مدل کم می‌شود؛ مانند ساخت هوش مصنوعی برای بازی شطرنج. الگوريتم‌های يادگيری ماشين در كاربردهای متنوعي مانند پزشكي، تشخيص گفتار، كشاورزی و بينايي كامپيوتری استفاده می‌شوند [95, 96].

## شبکه‌های عصبی

امروزه هوش مصنوعی، بسیاری از پیشرفت‌های خود را مدیون شبکه‌های عصبی است. این شبکه‌ها ساختاری مشابه مغز انسان دارند [97] و به دلیل پیچیده بودن آنها، توانایی پیاده‌سازی آنها تنها توسط رایانه‌ها ممکن است. امروزه این شبکه‌ها شامل میلیون‌ها و حتی میلیارد‌ها متغیر هستند که همواره در فرایند آموزش آنها در حال تغییر و به روزرسانی هستند. این شبکه‌ها شامل سه لایه ورودی، لایه‌ پنهان و لایه خروجی هستند. لایه‌های ورودی شامل گره‌هایی هستند که داده‌های آموزشی به آنها داده می‌شود. تعداد گره‌ها در این لایه باید با متغیر‌های ورودی برابر باشد؛ به طور مثال اگر تصویر به عنوان ورودی به شبکه می‌دهیم، باید تعداد گره‌ها برابر با وضوح تصویر ضرب در 3 باشد (تعداد کانال‌های تصویر را باید در نظر گرفت) و اگر ورودی ما مجموعه‌ای از اعداد باشد، باید به تعداد اعداد گره داشته باشیم. همچنین به منظور ساده‌سازی محاسبات، این ورودی‌ها باید تبدیل به اعدادی اعشاری[[38]](#footnote-38) بین 0 و 1 بشوند. لایه دوم، لایه پنهان نام دارد که می‌تواند تنها یک و یا چندین لایه از گره‌ها داشته باشد. این لایه، ورودی‌ها را از لایه قبل گرفته و اعمال محاسباتی را انجام می‌دهد. پس از انجام محاسبات، نتایج به‌دست‌آمده از این لایه، به لایه بعدی انتقال‌داده می‌شود. نام لایه آخر، لایه خروجی است و در صورتی که نوع یادگیری مورد استفاده، یادگیری با نظارت و یا نیمه‌ نظارت باشد، تعداد گره‌ها با تعداد طبقه‌ها برابر خواهد بود، به طوری که اگر 6 طبقه داشته باشیم، تعداد گره‌ها نیز 6 خواهد بود. پس از انجام محاسبات توسط لایه پنهان، اعداد حاصل‌شده به لایه خروجی انتقال می‌یابند و تصمیم گرفته می‌شود که این ورودی به کدام خروجی تعلق دارد. شکل 3-1 یک شبکه عصبی بسیار ساده را نشان می‌دهد. همچنین شبکه‌ای که لایه پنهان نداشته باشد، شبکه پرسپترون[[39]](#footnote-39) نام دارد که ساده‌ترین ساختار شبکه‌ی عصبی است و در شکل 3-2 آمده است.

|  |
| --- |
|  |
| شکل 40- ساختار یک شبکه عصبی با 2 لایه مخفی |

### روش پیش‌بینی شبکه

همانطور که گفته شد، شبکه پرسپترون ساده‌ترین شبکه عصبی است که روش پیش‌بینی آن به صورت زیر است: در ابتدا شبکه اعداد مربوط به ورودی‌ها را که بین 0 و 1 می‌باشد از لایه اول گرفته و و هرکدام را با بایاس[[40]](#footnote-40) آن گره جمع کرده و سپس در وزن مربوطه (wk)ضرب کرده و در نهایت تمامی این حاصل ضرب‌ها را جمع می‌کند. مجموع این حاصل ضرب‌ها سپس وارد یک تابع فعال‌ساز می‌شود (در اینجا تابع فعال‌ساز ما تابع پله‌ای[[41]](#footnote-41) است). این تابع برای‌ ما خروجی 0 یا 1 تولید می‌کند. در صورتی که این مجموع بیشتر یا مساوی صفر باشد، تابع فعال‌ساز عدد 1 و در صورتی که کمتر از صفر باشد، تابع فعال‌ساز به ما خروجی 0 را خواهد داد. از این شبکه برای حل مسائل طبقه‌بندی غیر خطی و بسیار ساده استفاده می‌شود [98]. ساختار این شبکه در شکل 3-2 نمایش داده شده. رابطه پیش‌بینی و خروجی شبکه پرسپترون و به طور کلی هر گره داخل شبکه در زیر آمده است. این رابطه از مدل مَک کولوه-پیتس[[42]](#footnote-42) به دست می‌آید [99].

|  |  |
| --- | --- |
| **(رابطه 3-1)** |  |
|  | |
| شکل 41- شبکه عصبی پرسپترون | |

### روش آموزش پس‌انتشار

در روش آموزش تحت نظارت، پس از پیش‌بینی شبکه‌ی عصبی از داده‌های ورودی و مشخص‌کردن‌ طبقه اختصاص داده شده به آنها، دو حالت به وجود می‌آید. حالت اول آن است که شبکه طبقه‌ درست را پیش‌بینی کرده و همان طبقه‌ای که در مجموعه‌داده برای آن داده‌ها مشخص کرده را ارائه داده است. حالت دوم آن است که شبکه طبقه‌ی نادرستی را برای داده‌های ورودی انتخاب کرده و با طبقه‌ای که مجموعه‌داده مشخص کرده فرق دارد. در فرایند آموزش یک شبکه‌عصبی تحت نظارت، باید وزن‌ها (wij) پس از هر خروجی به روزرسانی شوند. برای این امر، از الگوریتم پس‌انتشار[[43]](#footnote-43) استفاده می‌شود. در ابتدای مدل‌سازی شبکه‌عصبی در رایانه، این وزن‌ها به صورت تصادفی انتخاب می‌شوند. پس از آن فرایند آموزش آغاز می‌شود و ورودی‌ها حرکت رو به جلو طبق رابطه 3-1 دارند تا خروجی مشخص شود. همواره ممکن است در حالت درست یا غلط بودن طبقه‌ی تعیین شده، مقداری خطا وجود داشته باشد. این مقدار خطا توسط تابع هزینه و با مقایسه طبقه‌های ارائه شده توسط مجموعه‌داده محاسبه می‌شود. پس از آن، این خطاها در لایه‌ی خروجی به سمت لایه‌های ورودی حرکت می‌کنند و از لایه‌های پنهان می‌گذرند. با استفاده از خطا‌های به دست آمده برای هر گره، بهینه‌ساز وزن‌ها را به روزرسانی می‌کند [100] به طوری که اگر دفعه بعد همان ورودی‌ها و یا مشابه آنها به شبکه داده‌شد، خروجی مشابه‌تری به طبقه داده شده تولید کند. نمای کلی روش پس‌انشتار در شکل 3-3 نمایش داده شده.

در کنار روش آموزش پس‌انتشار، معمولا به شبکه‌های عصبی، داده‌های ورودی را به صورت دسته‌ای می‌دهد. در صورتی که داده‌ها را به صورت تکی به شبکه‌‌ها دهیم، فرایند آموزش بسیار زمان می‌برد. در صورتی که کل داده‌ها را به صورت یکجا به شبکه عصبی دهیم، ممکن است همه‌ی آنها بر روی حافظه سخت افزار جا نشوند. بنابراین جهت آموزش و اعتبارسنجی، داده‌ها را به صورت دسته‌هایی با ابعاد متناسب در می‌آیند و به شبکه داده می‌شوند. همچنین این امر باعث بهترشدن نتیجه آموزش و بالارفتن دقت می‌شود.

|  |
| --- |
|  |
| شکل 42- روش آموزش پس‌انتشار |

## شبکه‌های عصبی پیچشی

شبکه‌های عصبی پیچشی، به طور خاص برای تحلیل و پردازش داده‌های دارای بصری مانند تصاویر معمولی، فیلم‌ها و تصاویر پزشکی به کار می‌روند. این شبکه‌ها، انقلابی در زمینه یادگیری عمیق هستند که روش‌های شناسائی تصاویر، شناسائی الگو، شناسائی اشیا و غیره را دگرگون کرده‌اند. آنها توانایی استخراج و یادگیری ویژگی‌های تصاویر به صورت خودکار را دارند که باعث پیشرفت هوش مصنوعی در صنایع مختلفی چون کشاورزی، خودروسازی و پزشکی شده‌اند. روش‌های سنتی یادگیری ماشین توانایی شناسائی ویژگی‌ها در تصاویر پیچیده را نداشتند. به طور مثال، شناسائی لبه‌ها، الگوها و جزئیات تصاویر که بسیار چالش‌برانگیز هستند، امکان مشخص شدن توسط مهندسین پردازش تصویر را ندارند و یا بسیار سخت این کار انجام می‌شود. شبکه‌های عصبی پیچشی این محدودیت‌ها را رفع می‌کنند و می‌توانند اشیاء و الگو‌ها را با دقت بسیار بالاتر شناسائی کنند.

ساختار شبکه‌های عصبی پیچشی از فرایندهای زیستی قشر بینایی گربه الهام گرفته‌شده‌است. این ساختار به گونه‌ای است که تک‌نورون‌ها تنها در یک ناحیه محدود به تحریک پاسخ می‌دهند که به آن ناحیه پذیرش گفته می‌شود [101]. به طور مشابه، این شبکه‌ها فیلترهایی برروی تصاویر ورودی اعمال می‌کنند تا ویژگی‌ها و الگوهای مدنظر را شناسائی شوند. شکل 3-4 نمونه‌ای از ساختار این شبکه‌ها نمایش داده شده. در این شبکه‌ها ماهیتی سلسله‌مراتبی وجود دارد که باعث می‌شود ویژگی‌ها و الگو‌های ساده‌تر در لایه‌های اولیه تشخیص داده شوند و به تدریج ویژگی‌های پیچیده‌تر در لایه‌های عمیق‌تر بیاموزند. یک شبکه عصبی پیچشی از یک لایه ورودی، لایه‌های پیچشی، لایه‌های ادغام[[44]](#footnote-44) ، لایه‌های نرمال‌سازی دسته[[45]](#footnote-45) ، لایه‌های کاملا متصل[[46]](#footnote-46) و لایه‌ی خروجی تشکیل شده است. در ادامه، به معرفی لایه‌های شبکه عصبی پیچشی می‌پردازیم.

|  |
| --- |
|  |
| شکل 43- یک نمونه شبکه عصبی پیچشی برای طبقه بندی تصاویر مربوط به اعداد، شامل 2 لایه پیچشی، دو لایه ادغام بیشینه و دو لایه کاملا متصل |

### لایه پیچشی

لایه‌های پیچشی مهمترین جزء شبکه‌های عصبی پیچشی بوده و بیشترین محاسبات در این لایه انجام می‌شود. این لایه ویژگی‌های تصویر را استخراج کرده و ترکیبی از عملیات خطی و یا غیر خطی از عملیات پیچشی و تابع فعالسازی است. این لایه‌ها توانایی استخراج ویژگی‌ها در ناحیه‌های کوچک دارند و شبکه به جای آن که کل سطح تصویر را مورد بررسی قرار دهد، می‌تواند الگوها را و ویژگی‌ها را به صورت مجزا تشخیص دهد. به همین دلیل این شبکه‌ها ابزاری قدرتمند در طبقه‌بندی تصاویر بوده و از شبکه‌های عصبی سنتی پیشی گرفته اند. دلیل نام گذاری این لایه به این اسم، استفاده از عملیات ریاضی به نام کانولوشن در این لایه می‌باشد [102]. در صورتی که تصویر، آرایه‌ای از اعداد باشد و یک کرنل[[47]](#footnote-47) (که خود آرایه‌ی بسیار کوچکتری بین 2\*2 یا 7\*7 باشد) داشته باشیم، عملیات کانولوشن به این صورت تعریف می‌شود: مبدأ مختصات در پردازش تصاویر، نقطه‌ی بالا سمت چپ است و در صورتی که به سمت راست یا پایین حرکت کنیم، شماره‌ی پیکسل‌ها در محور‌ها افزایش می‌یابد. در ابتدا، خانه‌ی بالا سمت چپ کرنل روی خانه‌ی بالا سمت چپ تصویر قرار می‌گیرد. سپس هرکدام از اعداد موجود در پیکسل‌های کرنل ضرب در اعدادی که روی پیکسل‌های تصویر قرار گرفته اند می‌شود و بعد از آن مجموع این حاصل ضرب‌ها با هم جمع شده و در یک پیکسل از آرایه‌ای جدید قرار داده می‌شوند. سپس این کرنل یک خانه به سمت راست حرکت کرده و دوباره این عملیات اجرا شده و یک پیکسل در سمت راست آرایه جدید پر می‌شود. پس از آن که پیکسل بالا سمت راست کرنل روی پیکسل بالا راست آرایه تصویر قرار می‌گیرد و عملیات ضرب و جمع انجام می‌شود، کرنل دوباره به سمت چپ تصویر رفته اما این بار یک سطر پایین‌تر می‌آید. پس از پایان عملیات برروی این سطر نیز دوباره به سطر بعدی می‌رود. این کار آنقدر ادامه پیدا می‌کند تا خانه‌ی پایین راست کرنل روی پیکسل پایین راست تصویر قرار بگیرد. پس از، آرایه جدید تکمیل شده که به آن نقشه ویژگی[[48]](#footnote-48) گفته می‌شود [103]. شکل 3-5، عملیات کانولوشن یا پیچش را نمایش می‌دهد. از آن جایی که تصاویر رنگی شامل سه متغیر قرمز، سبز و آبی (RGB) و یا متغیرهای دیگر در فضاهای رنگی دیگر هستند، تصاویر ورودی سه آرایه برای انجام این عملیات دارند. بنابراین، در ابتدا سه نقشه ویژگی تولید می‌شود. عملیات کانولوشن روی نقشه ویژگی‌ها نیز انجام می‌شود و نقشه ویژگی‌های جدید تولید خواهد شد و این مرحله ممکن است چندین بار توسط کرنل‌های با ابعاد مختلف تکرار شود. این روش، دلیل یادگیری ویژگی‌های مهم مانند لبه‌ها، الگوها و بافت‌ها توسط شبکه‌های عصبی پیچشی است.

|  |
| --- |
|  |
| شکل 44- نمونه‌ای از عملیات کانولوشن |

### لایه تابع فعالسازی

بعد از لایه‌های پیچشی و یا لایه‌های پنهانی تمام‌متصل، توابع فعال سازی قرار می‌گیرد. این توابع وظیفه مهمی در شبکه‌های عصبی دارند. آنها به شبکه‌ها قابلیت استفاده از توابع و تبدیلات غیرخطی داده و توانایی شناسائی ارتباط‌های پیچیده بین ویژگی‌ها را فراهم می‌کند، به طوری که این شبکه‌ها قادر می‌شوند ویژگی‌های پیچیده‌تر را یادگرفته و پیش‌بینی‌های دقیق‌تری انجام دهد. بدون وجود توابع فعالسازی غیر خطی، شبکه عصبی به استفاده از ویژگی‌ها به صورت خطی محدود شده و به میزان قابل توجهی توانایی شناسائی داده‌های پیچیده را از دست می‌دهد. بنابراین، اضافه کردن قسمتی غیر خطی به شبکه‌ها امری مهم بوده و بر عهده توابع فعالساز غیر خطی می‌باشد. در ادامه به معرفی توابع غیرخطی پرکاربرد می‌پردازیم.

#### تابع فعالسازی یکسوساز

این تابع[[49]](#footnote-49) ، پرکاربردترین تابع فعالی سازی غیرخطی است. این تابع ورودی‌های مثبت را به صورت خطی نگاشت می‌کند و مقادیر منفی را برابر با صفر می‌کند. اگرچه این تابع، ظاهری شبیه به تابع خطی دارد، اما این تابع مشتق‌پذیر است و از آن می‌توان در مرحله پس‌انتشار استفاده کرد. این تابع تمامی گره‌ها را همزمان فعال نمی‌کند. به عبارتی، گره‌ها زمانی غیرفعال هستند که مقدار ورودی این تابع کمتر از صفر باشد و براساس نمودار این تابع، چنانچه ورودی تابع بزرگتر از صفر باشد، خروجی تابع برابر با ورودی تابع و اگر ورودی تابع، مقداری کم‌تر از صفر باشد، خروجی تابع برابر با صفر است [104, 105]. محدودیت مهمی که تابع یکسوساز با آن مواجه می‌شود، مسئله مرگ[[50]](#footnote-50) در این تابع است. این حالت زمانی اتفاق می‌افتد که خروجی چندین گره شبکه از تابع به صورت متوالی برابر با صفر باشد. رابطه 3-2 و شکل 3-6-الف مربوط به این تابع است.

|  |  |
| --- | --- |
| **(رابطه 3-2)** | ***f(x) = max(0 , x)*** |

#### تابع فعالسازی سیگموید

این تابع فعالسازی غیر خطی، ورودی خود را به مقادیری در بازه 0 و 1 تبدیل می‌کند. رابطه 3-3 در زیر نشان‌دهنده این تابع است. در صورتی که ورودی 0 به تابع فعالسازی سیگموید[[51]](#footnote-51) دهیم، خروجی 0.5 را تولید می‌کند. هر چه مقدار ورودی بیشتر باشد، خروجی به 1 نزدیکر (به ازای عدد 6، خروجی 0.997 تولید می‌شود) و هر چه ورودی کمتر باشد، خروجی به 0 نزدیکتر ( به ازای عدد 6-، خروجی 0.002 تولید می‌شود) خواهد بود. از ویژگی‌های تابع سیگموید می‌توان به این اشاره کرد که تغییرات بزرگ ورودی این تابع، باعث ایجاد تغییرات کوچک در خروجی می‌شوند [104, 106]. ایراد اصلی آن هم همین می‌باشد که برای مقادیر گرادیان بالا، تغییرات کمی اتفاق می‌افتد و باعث به وجود آمدن مسئله محوشدگی گرادیان[[52]](#footnote-52) می‌شود. رابطه 3-3 و شکل 3-6-ب، رابطه و نمودار این تابع است.

|  |  |  |
| --- | --- | --- |
| **(رابطه 3-3)** |  | |
|  | |  |
| الف) یکسوساز خطی | | ب) سیگموید |
| شکل 45- نمودار تابع فعالسازی الف) یکسوساز خطی و ب) سیگموید | | |

#### تابع فعالسازی سوویش

تابع فعالسازی سوویش[[53]](#footnote-53) توسط محققان گوگول ارائه شده و از لحاظ کارایی، عملکرد بهتری نسبت به تابع فعالسازی یکسوساز دارد. در این تابع، زمانی که مقدار x به سمت منفی بی‌نهایت میل می‌کند، مقدار y به سمت مقداری ثابت نزدیک می‌شود. با میل کردن مقدار x به مقدار مثبت بی‌نهایت، مقدار y نیز به سمت مثبت بی‌نهایت میل می‌کند. شیب منحنی تابع سوویش ملایم است و مانند تابع فعالسازی یکسوساز زمانی که مقدار x نزدیک به صفر است به یکباره تغییر جهت نمی‌دهد. به عبارتی، منحنی این تابع به آرامی از مقدار 0 به مقدار کوچک‌تر از 0 و پس از آن به مقادیر بیشتر از 0 تغییر جهت می‌دهد [104, 105]. در تابع فعالسازی یکسوساز، مقادیر منفی به عدد 0 تبدیل می‌شدند؛ در حالی که این مقادیر منفی ممکن است شامل اطلاعات مهمی از الگوهای داده‌ها باشند. از این تابع در معماری شبکه‌های Efficientnet [88] استفاده شده است که در ادامه به آنها خوهیم پرداخت.

|  |  |
| --- | --- |
| **(رابطه 3-4)** |  |

#### تابع فعالسازی بیشینه هموار

ایده طراحی تابع بیشنیه هموار نیز از تابع فعالسازی سیگموید گرفته شده است. براساس خروجی تابع سیگموید که بین بازه 0 تا 1 است، از آن تابع برای محاسبه احتمال استفاده می‌شود. با این حال، تابع فعالسازی سیگموئید یک مشکل اساسی دارد. فرض کنید در لایه آخر میخواهیم تابع سیگموید قرار دهیم. چنانچه چندین خروجی به صورت 0.8، 0.9، 0.6 و 0.7 تولید شوند، جمع تمامی خروجی‌ها بیشتر از عدد 1 می‌شود؛ در حالی که جمع مقادیر احتمالاتی باید عدد 1 باشد. تابع فعالسازی بیشنه هموار، نوعی از تابع سیگموئید به حساب می‌آید. به عبارتی، این تابع ترکیبی از چندین تابع سیگموئید است که احتمالات نسبی برای چندین طبقه را محاسبه می‌کند [105]. از این تابع در لایه آخر برای مسائل طبقه‌بندی چندطبقه استفاده می‌شود. در رابطه 3-5 و شکل 3-7-ب، رابطه و نمودار تابع آمده است.

|  |  |  |
| --- | --- | --- |
| **(رابطه 3-5)** |  | |
|  | |  |
| الف) سوویش | | ب) بیشینه هموار |
| شکل 46- نمودار تابع فعالسازی الف) سوویش و ب) بیشنیه هموار برای 2 متغیر | | |

### لایه پَدینگ

هدف عملیات پیچش، استخراج ویژگی‌های سطح بالا مانند لبه‌ها از تصاویر ورودی است. شبکه‌های عصبی پیشچی، نیاز دارند که صرفا به یک لایه پیچشی محدود نشود. به طور معمول، اولین لایه پیچشی، مسئول ثبت ویژگی‌های سطح پایین مانند لبه‌ها، رنگ، جهت گرادیان و دیگر موارد است. با افزودن لایه‌های بیشتر، مدل با ویژگی‌های سطح بالا نیز سازگار می‌شود و شبکه‌ای به‌دست می‌آید که دارای درک کاملی از تصاویر موجود در مجموعه‌داده به صورتی است که انسآنها تصاویر را درک می‌کنند .اما استفاده مکرر از این لایه‌ها و عملیات کانولوشن، باعث کوچکترشدن تصاویر و نقشه ویژگی‌ها می‌شوند، چرا که تعداد پیکسل‌های تولید شده در آرایه جدید، یک یا چند تا از سطرها و ستون‌های کناریشان حذف می‌شود. برا حل این مشکل از لایه پَدینگ[[54]](#footnote-54) استفاده می‌شود. این لایه انواع مختلفی دارد، اما رایج‌ترین آن می‌تواند اطراف تصویر یا نقشه ویژگی ورودی، یک یا چند سطر و ستون اضافه کند تا هنگام انجام عملیات کانولوشن، ابعاد آرایه خروجی با ابعاد آرایه ورودی برابر باشد. شکل 3-8، کارکرد لایه پدینگ در عملیات پیچش را نشان می‌دهد.

|  |
| --- |
|  |
| شکل 47- نمونه‌ای از کاربرد لایه Padding برای حفظ ابعاد تصویر |

### لایه ادغام

همانند لایه پیچشی، لایه ادغام نیز مسئول کاهش سایز فضای ویژگی‌ها و نقشه‌ ویژگی‌ها می‌باشد. این کار با هدف کاهش قدرت محاسباتی لازم برای پردازش داده‌ها از طریق کاهش ابعاد انجام می‌شود. لايه ادغام معمولاً پس از چندين لايه پيوسته كانولوشني اضافه مي‌شود. دو نوع لایه ادغام پرکاربرد وجود دارد. شکل 3-9 این لایه را نشان می‌دهد.

* **لایه ادغام بیشینه[[55]](#footnote-55) :** این لایه، پر استفاده‌ترین لایه ادغام است. عملیات این لایه به این صورت است که تصاویر و نقشه ویژگی‌ها را به مجموعه مربع‌هایی به ابعاد 2 در 2 و یا ابعاد دیگر تقسییم کرده و از بین پیکسل‌های هرکدام از این مربع‌ها، بیشترین مقدار موجود را برداشته و مربعی جدید با آنها تولید می‌کند. لایه ادغام ماکزیمم کار حذف نویز را نیز انجام می‌دهد. بسته به پیچدگی تصاویر، تعداد این لایه‌ها ممکن است برای ثبت جزئیات سطح پایین بیشتر افزایش یابد.
* **لایه ادغام میانگین[[56]](#footnote-56) :** این لایه، به جای آن که حداکثر مقدار را از میان مستطیل‌های رسم شده انتخاب کند، میانگین پیکسل‌های آنها را گرفته و در نقشه ویژگی تولیدشده قرار می‌دهد. معمولا فقط یک بار از این لایه، آن هم قبل از لایه‌های کاملا متصل استفاده می‌شود. به همین دلیل نام دیگر این لایه، لایه ادغام میانگین کلی[[57]](#footnote-57) نیز می‌باشد.

|  |
| --- |
|  |
| شکل 48- نمونه‌ای از عملکرد لایه‌های ادغام بیشنیه و میانگین |

### لایه نرمال‌سازی دسته

آموزش شبکه عصبی عمیق به واسطه پراکندگی داده‌های خروجی از لایه و ورود به لایه بعد، نیازمند کاهش نرخ یادگیری و تنظیم دقیق پارامتر‌های اولیه هر لایه است. این امر موجب کاهش سرعت آموزش شده و فرایند آموزش چندین نیازمند تعداد دوره‌های بیشتری برای همگرایی دارد. در لایه نرمال سازی دسته که روی ورودی هر لایه شبکه عصبی عمیق اعمال می‌شود، تکنیکی اعمال می‌شود که از طریق تغییر مرکز توزیع داده‌ها یا تغییردادن مقیاس آنها موجع سریعتر شدن شبکه عصبی می‌شود. این لایه واریانس هر دسته بر روی 1 و میانگین آنها را روی 0 تنظیم می‌کند. شبکه‌هایی که از این لایه بهره می‌برند، از مشکل محوشدگی گرادیان در حین پس‌انتشار کمتر رنج می‌برند [107].

### لایه برون‌انداز

کار لایه برون‌انداز[[58]](#footnote-58) ، غیر فعالسازی برخی از نورون‌های یک لایه است. در واقع پس از هر لایه‌ای که لایه برون‌انداز قرار بگیرد، با توجه به عدد احتمالی که دارد، هر گره‌ی شبکه ممکن است غیرفعال و مقدار آن صفر شود. در حالت عادی ممکن است شبکه تنها برخی از ویژگی‌های خاص استخراج شده را برای طبقه بندی استفاده کند که موجوب پیش آمدن پدیده بیش برآزش[[59]](#footnote-59) می‌شود. ام این لایه به آموزش شبکه با استفاده از کل ویژگی‌های بدست آمده از داده‌های ورودی کمک می‌کند [108]. در شکل 3-10 نحوه عملکرد این لایه نمایش داده شده است.

|  |  |  |
| --- | --- | --- |
|  | | |
| الف) عدم وجود لایه برون‌انداز |  | ب) وجود لایه برون‌انداز |
| شکل 49- مقایسه عملکرد شبکه در حین آموزش الف) بدون لایه برون‌انداز ب) با لایه برون‌انداز | | |

#### لایه بلوک برون‌انداز

این لایه بسیار شبیه لایه برون‌انداز می‌باشد، با این تفاوت که به جای یک گره، مجموعه‌ای از گره‌ها را غیر فعال می‌کند. اگر فرض شود تصویری وجود دارد که بر روی آن عملیات کانولوشن انجام شده است، لایه برون‌انداز تنها تعدادی از پیکسل‌های به وجود آمده در نقشه ویژگی را غیر فعال می‌کند. اگر از لایه بلوک برون‌انداز[[60]](#footnote-60) استفاده شود، بسته به احتمال داده شده و سایز بلوک k، یک مربع کامل از نقشه ویژگی‌های تولید شده حذف می‌شود. یعنی یک نقطه به مرکزیت انتخاب شده و پبکسل‌های اطراف آن غیرفعال و 0 می‌شوند. استفاده از این تکنیک به یادگیری بهتر شبکه و افزایش دقت کمک می‌کند [109]. در شکل 3-11، مقایسه‌ای بین عملکرد دو لایه برون‌انداز و بلوک برون‌انداز برروی نقشه ویژگی‌ها نایش داده شده است.

### لایه کاملا متصل

این لایه، از آخرین لایه‌های شبکه‌های عصبی می‌باشد و قبل از تابع فعالسازی آخرین لایه می‌باشد، می‌آید. لایه کاملا متصل[[61]](#footnote-61) در شبکه‌های پیچشی نقش طبقه‌بند را دارند. همچنین این امکان وجود دارد که پس از آن که استخراج ویژگی‌ها توسط لایه‌های پیچشی و نمونه‌برداری‌ها توسط لایه‌های ادغام انجام می‌شود، چندین لایه کاملا متصل با تعداد گره‌های متفاوت به همراه توابع فعالسازی آنها پشت سر هم قرار داده شوند تا شبکه طبقه‌بندی بهتری داشته باشد. هدف اصلی استفاده از این لایه، یادگیری ویژگی‌های کل داده‌های ورودی است.

|  |  |
| --- | --- |
|  | |
| الف) تاثیر لایه برون‌انداز | ب) تاثیر لایه بلوک برون انداز |
| شکل 50- مقایسه روش‌های برون‌اندازی و خروجی شده توسط الف) لایه برون‌انداز معمولی ب) لایه بلوک برون‌انداز [109] | |

### تابع فعالسازی آخرین لایه

پس از آخرین لایه کاملا متصل، تابع فعالسازی آن می‌باشد. وظیفه این قسمت، اعلام خروجی نهائی شبکه عصبی می‌باشد. به طور کلی در مسائل طبقه‌بندی، اگر به صورت چندطبقه باشد، از تابع فعالسازی بیشینه هموار استفاده می‌شود. همانطور که گفته شد، این تابع به صورت احتمال عمل می‌کند و پس از دریافت اعداد ورودی، در خروجی، جمع آنها را برابر 1 می‌کند و همه‌ی طبقه‌ها احتمالاتی بین 0 و 1 دریافت می‌کنند. طبقه‌ای که بیشترین احتمال را داشته باشد، به عنوان طبقه برنده و خروجی انتخاب می‌شود. در مسائلی که به صورت چند برچسبه و یا دوتایی[[62]](#footnote-62) هستند و هر داده‌ی ورودی ممکن است به چند طبقه تعلق داشته باشند، معمولا از تابع فعالسازی سیگموید در آخرین لایه استفاده می‌شود. این تابع با نگاشت ورودی‌های خود بین 0 و 1 در خروجی امکان حضور چندین طبقه را فراهم می‌سازد، به طوری که بتوان گفت اگر عدد خروجی مربوط به هر طبقه از یک آستانه‌ای بیشتر باشد، داده‌ی ورودی شامل آن طبقه می‌باشد.

## معماری‌های شبکه‌های عصبی پیچشی

همانطور که قبلا گفته شد، در سال‌های اخیر پردازش تصویر به لطف پیشرفت سخت افزار‌ها و شبکه‌‌های عصبی پیچشی به دقت‌های بسیار بالا در زمینه طبقه‌بندی تصاویر رسیده است. در ادامه به معرفی برخی از شبکه‌های عصبی پیچشی معرفی شده در سال‌های اخیر پرداخته شده است.

### معماری Resnet و Resnext

شبکه عصبی پیچشی Resnext، در واقع برگرفته شده از شبکه عصبی پیچشی Resnet است. شبکه‌ی Resnet در سال 2015 در تحقیق [89] معرفی شد و برند‌ه‌ی چالش ImageNet آن سال می‌باشد. تا به امروز این شبکه از موفق‌ترین‌ها در امر دسته‌بندی تصاویر بوده است. به طور کلی، در شبکه‌های عصبی عمیق هرچه‌لایه‌های بیشتری را اضافه کنیم، این شبکه‌ها عمیق‌تر شده و ویژگی‌های پیچیده‌تری را یاد می‌گیرند اما اضافه کردن لایه‌های بیشتر ممکن است عملکرد شبکه‌ی عصبی را تخریب کند و میزان خطای آن را افزایش دهد. دلیل آن، به وجود آمدن مشکل محو شدگی گرادیان است. این مشکل در شبکه‌های پیچشی قدیمی‌تر به وضوح قابل مشاهده‌ است. در معماری Resnet اتصالاتی میان‌بر وجود دارد که باعث می‌شود شبکه از یک یا چند لایه عبور کند و لایه‌ها را به طور مستقیم به لایه‌های دورتر متصل کند. این شبکه از معماری VGG الهام گرفته شده است اما با استفاده از این تکنیک و بلوک‌های میانبر می‌تواند تا 150 لایه نیز داشته باشد [89]. ساختار بلوک میانبر در شکل 3-12 نمایش داده شده است.

|  |
| --- |
|  |
| شکل 51- ساختار بلوک میانبر در معماری شبکه Resnet [90] |

پس از معرفی شبکه‌ی Resnet، تحقیقاتی جهت بهبود این ساختار منتشر شدند. یکی از این پژوهش‌ها، [110] بود که ساختاری تحت عنوان Resnext ارائه داد و در چالش ImageNet سال 2016 مقام دوم را کسب کرد. در این معماری از بلوک‌هایی استفاده شده که مجموعه‌ای از تبدیلات را با همان معماری اصلی ترکیب می‌کند. این طراحی معماری چند شاخه‌ای دارد که تعداد ابرپارامترهای لازم برای آموزش شبکه را کاهش می‌دهند. در این استراتژی از پارامتر کاردینالیته[[63]](#footnote-63) (ابعاد مجموعه تبدیلات) استفاده شده است که با افزایش مقدار آن، دقت طبقه‌بندی تصاویر و حتی تصاویر پیچیده نیز بالاتر می‌رود. علاوه بر آن، افزایش کاردینالیته موثرتر از افزایش ابعاد شبکه است [110]. شکل 3-13 و شکل 3-14 به بررسی تفاوت ساختار دوشبکه Resnet و Resnext پرداخته اند.

|  |  |
| --- | --- |
|  | |
| الف) بلوک ساختار Resnext | ب) بلوک ساختار Resnet |
| شکل 52- مقایسه ساختارهای بلوک‌های سازنده الف) شبکه Resnext با کاردینالیته 32 و ب) شبکه Resnet با یک دیگر. هر دو بلوک پیچیدگی تقریبا یکسانی دارند و اتصالات میانبر در آنها رسم شده است [110] | |
|  | |
| شکل 53- مقایسه‌ی بین ساختار شبکه‌های Resnet50 و Resnext50 در قالب 32x4d. میزان کاردینالیته=32 نشان‌دهنده تجمع کانولوشن‌ در 32 گروه است [110] | |

### معماری Efficientnet

شبکه‌های عصبی پیچشی معمولا بر پایه یک سخت‌افزار ثابت طراحی می‌شوند و درصورت امکان، اگر منابع سخت‌افزاری بیشتری موجود بود، شبکه را پیچیده‌تر و عمیق‌تر می‌کنند تا دقت بالاتری حاصل شود. در مقاله [88] که در سال 2019 منتشر شد، به بررسی تاثیر عرض، عمق و وضوح تصاویر ورودی بر دقت طبقه‌بندی پرداخته شد. بر این اساس ضریبی به نام ترکیب[[64]](#footnote-64) ارائه شد که همه پارامتر‌های مطرح شده را در مقیاس‌ها و ابعاد مختلف در کنار هم قرار می‌دهد. تاثیر این ضریب را برروی شبکه‌های خانواده Resnet و Mobilenet آزمایش شد و افزایش دقت در طبقه‌بندی مشاهده شد. مقایسه افزایش این متغیر‌ها در شکل 3-15 نمایش داده شده است.

برای استفاده بهتر از این تکنیک، خانواده‌ای جدید از شبکه‌های عصبی پیچشی طراحی شد که Efficientnet نام گرفته شدند که عمیق‌ترین آنها، افزایش دقت قابل توجهی نسبت به باقی خانواده‌های شبکه‌های عصبی داشتند. همچنین ابعاد این شبکه بسیار کاهش داشته و سرعت آن نیز تا 6 برابر سریعتر شده است. در این تحقیق از الگوریتم‌های بهینه‌سازی برای یافتن عرض، عمق و وضوح تصاویر استفاده شده تا بهترین و کوچکترین ترکیب ممکن برای شبکه عصبی پیچشی در زمان خود شناسائی شود. آنها شبکه ابداع شده را EfficientnetB0 نامیدند و ساختار آن نیز در شکل 3-16 نمایش داده شده. پس از آن، با استفاده از ضریب‌های ترکیب مختلف، ابعاد این شبکه را افزایش دادند و شبکه‌های جدیدتری به نام EfficientnetB1 تا EfficientnetB7 ساختند که هر کدام دقت بالاتری نسبت به شبکه‌های هم نوع خود ارائه دادند [88]. پس از این تحقیق، به دنبال بررسی ایرادات این شبکه‌ها، نویسندگان آن در تحقیق دیگری [111] شبکه‌های عصبی خانواده EfficientnetV2 را معرفی کردند که هم ابعاد کمتری و هم سرعت بالاتر دارند. شبکه‌های Efficientnet به میزان یکسان و با استفاده از یک عدد ترکیب ابعاد شبکه‌ها و وضوح تصاویر ورودی را افزایش می‌دهند. به طور مثال اگر ضریب ترکیب 2 باشد، همه‌ی قسمت‌های شبکه دو برابر می‌شوند. اما این قسمت‌ها لزوما به یک میزان به سرعت آموزش شبکه و بازدهی افزایش پارامترها کمک نمی‌کنند. به علاوه این شبکه‌ها ابعاد تصاویر ورودی را افزایش می‌دهند، که باعث سرعت محاسبه پایین‌تر و حافظه بیشتر می‌باشد. برای حل این مشکلات، ابعاد تصاویر را به میزان حداقل نگه داشته شدند و تغییرات کوچکی در ضریب ترکیب داده شد. فرضيه موجود در اين مقاله به اين صورت بود كه، براي جلوگيري از بیش‌برآزش عكس‌هاي كوچك، تنوع‌سازی[[65]](#footnote-65) كمتر و عكس‌هايي كه بزرگ‌تر هستند نياز به تنوع‌سازی بیشتری دارند. با افزايش سايز مدل، هر چقدر ميزان تنوعات بيشتر باشد، دقت به دست آمده نيز بيشتر است. بنابراين، شيوه‌ي استفاده شده در تحقیق یاد شده، به اين صورت مي‌باشد كه به ازاي تعداد دوره‌های آموزشی موجود، ابعاد عكس و شدت تنوعات در هر دوره افزايش پيدا مي‌كند. نام این روش، آموزش پیشرفتی[[66]](#footnote-66) است. نتایج حاصل شده نشان می‌دهد که دقت شبکه EfficientnetV2M تقریبا برابر با EfficientnetB7 می‌باشد با این تفاوت که آموزش شبکه با استفاده از منابع سخت افزاری یکسان بسیار سریعتر شده است [111].

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
|  |  |  |  |  |
| الف) پایه | ب) افزایش عرض | ج) افزایش عمق | د) افزایش وضوح | ه) افزایش ترکیبی |
| شکل 54- روش مقیاس‌بندی در حین طراحی مدل‌های شبکه‌ی عصبی پیچشی. الف) مدل اصلی ب) افزایش عرض ج) افزایش عمق د) افزایش وضوح ورودی و ه) افزایش همه پارامتر‌ها به صورت ترکیبی [88] | | | | |
|  | | | | |
| شکل 55- ساختار شبکه EfficientnetB0 [88] | | | | |

### معماری SWin

در سال 2020 روشی جدا و متفاوت نسبت به شبکه‌های عصبی پیچشی برای طبقه‌بندی تصاویر معرفی شد [91]. در این معماری از لایه‌های مبدل برای تصاویر استفاده شده است. در پردازش زبان طبیعی[[67]](#footnote-67) از مبدل‌ها[[68]](#footnote-68) برای پردازش کلمات و جملات استفاده می‌شود، اما مبدل‌ها تا قبل از این تحقیق استفاده‌ی محدودی در پردازش تصویر داشته‌اند. در زمینه تصاویر، توجه در مدل‌های یادگیری عمیق یا توسط شبکه‌های عصبی پیچشی اعمال می‌شوند و یا توسط برخی از قسمت‌های تغییر یافته از لایه‌های پیچشی که ساختار کلی آنها حفظ می‌شود. در تحقیق ذکر شده، نشان داده شد که لزوما نیازی به شبکه‌های عصبی پیچشی برای طبقه‌بندی تصاویر نمی‌باشد و در صورتی که تصاویر را به صورت کلمات 16 در 16 در نظر گرفته شوند، می‌توان مبدل‌ها را روی این تصاویر که به چندین بخش[[69]](#footnote-69) تبدیل شده اند، اعمال کرد. نام این شبکه ViT یا مبدل بصری گذاشته شد که ساختار آن در شکل 3-17 نمایش داده شده. از جمله ایرادات این شبکه‌ها، نیاز به آموزش آنها بر روی مجموعه‌داده‌ای بسیار وسیع می‌باشد، به طوری که بین 14 الی 300 میلیون تصویر نیاز دارند. برای استفاده از روش یادگیری انتقالی این نوع شبکه‌ها نیز لازم است که مجموعه‌داده مورد نظر تعداد تصاویر زیادی داشته باشد، به طوری که مجموعه‌داده ImageNet خود از یادگیری انتقالی استفاده می‌کند [91]. در بخش 3-5 به بحث یادگیری انتقالی پرداخته می‌شود.

اما در سال 2021 و در تحقیقی دیگر نوعی دیگر از مبدلات تصاویر معرفی شدند. نام مدل طراحی شده بر این اساس، مبدل پنجره‌ای جابه‌جا شده[[70]](#footnote-70) می‌باشد. این مبدلات از روش تقسیم‌بندی به صورت سلسله‌مراتبی[[71]](#footnote-71) استفاده می‌کنند که بازدهی بالاتری دارد. این افزایش عملکرد به دلیل محدود کردن توجه بر روی پنجره‌هایی که با یک دیگر تماس ندارند و در عین حال اجازه ارتباط بین پنجره‌ای وجود دارد، هست. دقت مدل‌های ارائه شده با استفاده از این پنجره‌ها بر روی مجموعه‌داده ImageNet به 82% تا 85% می‌رسد [90]. معماری و تفاوت عملکرد شبکه SwinT در شکل‌های 3-18 و 19-3 نشان داده شده است.

|  |  |  |
| --- | --- | --- |
|  | | |
| شکل 56- معماری شبکه ViT. تصاویر به چند بخش با وضوح ثابت تقسیم شده و هر کدام از آنها به صورت داده‌های خطی در می‌آیند و هر بخش تصویر در کنار موقعیت قرارگیری آن، به یک مبدل استاندارد داده می‌شوند [112] | | |
|  | | |
| الف) شبکه ViT | ب) شبکه SwinT | |
| شکل 57- مقایسه عملکرد بین دو شبکه‌ی الف) ViT که مبدل ارائه شده نقشه ویژگی‌هایی با وضوح کمتر تولید می‌کند و ب) SwinT که مبدل آن، نقشه ویژگی‌ها به صورت سلسله‌مراتبی و با وضوح بالاتر می‌سازد [90] | | |
|  | |
| شکل 58- نمایش معماری شبکه Swin Transformer [90] | |

### معماری Regnet

شبکه‌های عصبی پیچشی به طور سنتی، برای یک هدف طراحی و بهینه‌سازی می‌شوند. به طور مثال، شبکه ResNet با هدف بیشترین دقت برروی مجموعه‌داده ImageNet در زمان خود طراحی شده بود. یا شبکه‌های EfficientNet برای انجام عملیات‌های بینایی رایانه به صورت بهینه طراحی شده بود و یا شبکه MobileNet برای اجرا بر روی تلفن‌های همراه. در سال 2020 و در مقاله‌ای تحت عنوان ((طراحی فضاهای طراحی شبکه)) [92] هدفی جدیدتر برای طراحی شبکه عصبی مناسب برای پردازش تصویر ارائه شد: طراحی ساختار شبکه‌ای چند منظوره که تمامی ویژگی‌های ذکر شده در بالا را داشته باشد. محصول نهایی، مجموعه شبکه‌های RegNet بود که دقتی برابر و سرعتی 5 برابر شبکه‌های EfficientNet داشتند.

فضای طراحی اولیه‌ای که برای انجام این فرایند در نظر گرفته شد، شبکه‌ی AnyNet بود که ساختار بسیار ساده و محدود‌شده‌ی شبکه ResNet است. ریشه‌ی آن لایه ورودی تصاویر بوده، بدنه آن قسمت اصلی محاسبات و هد (head) آن نیز بخش طبقه‌بندی می‌باشد. بدنه آن 4 مرحله و هر مرحله نیز تعداد متفاوتی از بلوک‌ها را دارد. شکل 3-20، جزئیات بیشتری از ساختار کلی شبکه نمایش می‌دهد.

|  |  |  |
| --- | --- | --- |
|  | | |
| الف) بلوک‌های مرحله i | ب) مرحله‌های بدنه | ج) ساختار کلی شبکه |
| شکل 59- ساختار کلی شبکه‌های طراحی شده RegNet الف) بلوک‌هایی موجود در مرحله i ب) مرحله‌های موجود در بدنه و ج) ساختار کلی شبکه اولیه معرفی شده [92] | | |

فضای طراحی AnyNetX، 16 درجه آزادی دارد، چرا که هر شبکه شامل 4 مرحله بوده و هر مرحله 4 پارامتر دارد: تعداد بلوک‌ها d، عرض بلوک w، نرخ گلوگاه[[72]](#footnote-72) b و عرض گروه g. با استفاده از یک رابطه خطی و انجام آزمایشات، محقیقن به طراحی نهائی این شبکه رسیدند که با توجه به قدرت محاسباتی مورد نیاز برای آموزش آنها، 12 شبکه اصلی از RegNetX200MF با قدرت محسباتی مورد نیاز 200 مگا فلاپ[[73]](#footnote-73) تا RegNetX32GF با قدرت محاسباتی مورد نیاز 32 گیگا فلاپ برای عملیات طبقه‌بندی مناسب است. در آزمایشی دیگر، به لایه‌های x لایه‌های Squeeze و Excitation اضافه شد و سری بهتری از این شبکه‌ها تحت عنوان RegNetY ارائه شد که اندکی بهتر از سری اصلی x می‌باشد (Y=X+SE). دقت این مدل‌ها به میزان قابل توجهی بالا بوده و قدرت محاسباتی کمی نیاز دارند [92]. جدول‌ 3-1 معماری‌های خانواده Regnet را مقایسه می‌کند.

|  |
| --- |
| جدول ‏3-1- مقایسه ساختار‌های معماری الف) RegnetX و ب) RegnetY با یکدیگر [92]. |
|  |
| الف) معماری‌های RegnetX |
|  |
| ب) معماری‌های RegnetY با فرض Y=X+SE. |

### معماری ConvNext

با پیدایش شبکه‌های پیچشی و به وجود آمدن انقلابی در پردازش تصویر، نقش محققین و مهندسین از طراحان استخراج‌کننده ویژگی‌ها به طراحان شبکه‌های عصبی تغییر یافت. در سال‌های اخیر در کنار پیشرفت شبکه‌*‌*های پیچشی، مسیر الگوریتم‌های پردازش زبان طبیعی نیز عوض شده است. به جای استفاده از شبکه‌های عصبی بازگشتی[[74]](#footnote-74) ، مبدل‌ها استفاده می‌شوند و نتیجه بهتری را ارائه می‌دهند. علی‌رغم تفاوت‌های اساسی میان پردازش زبان طبیعی و پردازش تصویر، در سال 2020 با معرفی مبدل‌های تصاویر این دو زمینه به یکدیگر برخورد کردند. درسال 2022 شبکه‌های خانواده ConvNext معرفی شدند که تولدی دوباره برای شبکه‌های عصبی پیچشی بود. طراحان این معماری، سعی داشته اند تا عملکرد و رفتار شبکه‌های Resnet50 را به مانند یک مبدل تصاویر تغییر دهند، بدون آن که به طور کلی به مبدل تصاویر تبدیل شود. آنها در ابتدا از تکنیک‌هایی که در فرآیند آموزش مبدل‌ها استفاده شده است، برای آموزش شبکه Resnet استفاده کردند. تکنیک‌هایی چون اضافه کردن تعداد دوره‌های آموزشی تا 300 دوره، تکنیک‌های داده‌افزایی مانند برش-ترکیب، مخلوط کردن[[75]](#footnote-75) ، پاک کردن تصادفی، نرم‌سازی برچسب‌ها و. . . باعث شدند که دقت این مدل فقط به واسطه تجدید نظر در روند آموزش آن 2.7% افزایش یابد.

|  |
| --- |
|  |
| شکل 60-نقشه راه ارائه شده در [113] برای ارائه شبکه‌ی پیچشی جدید |

پس از آن محققین با استفاده از تغییرات اعمال شده در هر مرحله (stage) و تبدیل ساختار ورودی بدنه (stem) و بخش‌بندی (patchify) کردن آن دقت مدل را افزایش دادند. در مرحله بعد، سعی شد تا با الهام گرفتن از ساختار شبکه Resnext، کانولوشن‌های ایجاد شده در مرحله قبل را به صورت گروهی در آورده شوند. در مرحله بعد، با توجه به این که مبدل‌های تصاویر، یک عنصر مهم به نام گلوگاه معکوس[[76]](#footnote-76) تولید می‌کنند، در این شبکه نیز قرار داده شد. در مرحله بعد، با افزایش سایز کرنل‌ها تا 7 در 7 دقت کمی بیشتر افزایش می‌یابد. در صورتی که سایز کرنل‌ها بیشتر از این شود، باعث کاهش دقت می‌شود. در مرحله بعد سعی شد تا با تغییر توابع فعالسازی از یکسوساز به یکسوساز خطی گوسی[[77]](#footnote-77) ، کاهش دادن تعداد توابع فعالسازی، کاهش تعداد لایه‌های نرمال‌سازی دسته و در نهایت تعویض لایه‌های نرمال‌سازی دسته با لایه‌های نرمال‌سازی لایه‌ای دقت افزایش یابد. نقشه راه این تحقیق در شکل 3-21 نشان داده شده. شبکه پیچشی تولید شده، دقت بالاتری نسبت به Swin Transformer از خود نشان می‌دهد و قدرت پردازش یکسانی جهت آموزش نیاز دارد. سپس این شبکه در 4 ابعاد مختلف ریز، کوچک، پایه و بزرگ ارائه داده شد [113]. شکل‌های 3-22 و 3-23 معماری و عملکرد شبکه‌های خانواده ConvNext را نمایش می‌دهند.

|  |  |
| --- | --- |
|  | |
| شکل 61- معماری شبکه ConvNext پایه (اتصال‌های میانبر نمایش داده نشده اند) | |
|  |
| شکل 62- مقایسه شبکه ConvNext با دیگر شبکه‌ها [113] |

### معماری WideResnet

شبکه‌ی Resnet یکی از محبوب‌ترین شبکه‌های پیچشی می‌باشد. پس از معرفی آن و به دست آوردن جایگاه اول در چالش ImageNet، تلاش‌های زیادی برای بهبود ساختار و عملکرد آن صورت گرفت. یکی از آنها، شبکه‌ی WideResnet بود که در [114] ارائه شد. در شبکه‌های Resnet، برای افزایش دقت می‌توان لایه‌ها را عمیق‌تر کرد، اما با این کار سرعت آموزش شبکه بسیار کاهش می‌یابد. برای حل این مشکل، سعی شده است در تحقیق ذکر شده به جای عمیق‌تر کردن لایه‌ها، عمق آنها را کاهش و عرض آنها را افزایش دهند. با این کار معماری عریض از این شبکه به وجود می‌آید . برای کاهش عمق شبکه و افزایش عرض آن، باید در ساختار بلوک‌های Residual تجدید نظر شود. به این منظور، سه پارامتر d، l و k به ترتیب که نشان‌دهنده‌ی تعداد بلوک‌ها، عمق آنها و عرض آنها هستند استفاده شده است. جهت بهنیه ‌سازی سرعت آموزش، مقدار d تقسیم بر k در نظر گرفته شد. اگر k=1 باشد، همان شبکه اصلی Resnet را ارائه می‌دهد. اگر هر عدد طبیعی دیگر باشد، آن لایه و شبکه k برابر عریض‌تر می‌باشد.

|  |  |  |  |
| --- | --- | --- | --- |
|  |  |  |  |
| الف) بلوک ساده | ب) بلوک bottleneck | ج) بلوک عریض ساده | ه) بلوک عریض با برون‌انداز |
| شکل 63- نواع بلوک‌های الف) ساده ب) bottleneck ج) عریض و ه) عریض با لایه برون انداز استفاده شده در شبکه WideResnet. بعد از هر لایه کانولوشن لایه‌های نرمال‌سازی دسته و تابع فعالسازی یکسوساز قرار گرفته است [114] | | | |

حالت‌های مختلف عمق و عرض آزمایش شدند و نتایج حاصل شده نشان می‌دهد که این مدل‌ها نسبت به شبکه مادر خود، Resnet، عملکرد مناسب‌تر و سریعتری داشته اند [114].

## یادگیری انتقالی

یادگیری عمیق، زیرمجموعه ای از یادگیری ماشین است که شامل انبوهی از الگوریتم‌های آموزشی می‌باشد. مدل‌های استفاده شده در زمینه یادگیری عمیق، بر پایه شبکه‌های عصبی عمیق هستند. مدل‌های شبکه‌ی عصبی عمیق توانایی استخراج ویژگی‌های مفید از تصاویر، متن، اعداد، صوت و. . . به شیوه‌های تحت نظارت و یا بدون نظارت دارند. به همین دلیل، نوع یادگیری آنها متمایز از روش‌های دیگر یادگیری ماشین است. با این ویژگی‌های یادگیری عمیق، استفاده از راه حل‌های یادگیری ماشین سنتی و به صورت دستی کمتر شده است.

یادگیری انتقالی یکی از تکنیک‌های یادگیری عمیق است. اگر مدلی وجود داشته باشد که وزن لایه‌های آن را از ابتدا و به صورت تصادفی انتخاب کرده باشیم و مدل را بر روی مجموعه‌داده‌ای آموزش دهیم تا مدل بتواند داده‌های جدیدتر را دسته‌بندی کند، به آن آموزش از ابتدا گفته می‌شود. برای این کار باید از مجموعه‌داده‌ای بزرگ و وسیع و تعداد دوره‌های آموزشی زیاد استفاده شود تا مدل به اندازه کافی آموزش ببیند و وزن لایه‌های آن به اعداد مناسبی همگرا شوند. به طور معمول، افرادی که شبکه‌های عصبی پیچشی برای طبقه‌بندی تصاویر دو بعدی ارائه می‌کنند، به منظور سنجش دقت و صحت شبکه طراحی شده آن را با استفاده از مجموعه‌داده‌ی استاندارد ImageNet همانند باقی شبکه‌های عصبی پیچشی آموزش می‌دهند [57]. این مجموعه‌داده در سال 2010 تحت چالش ILSVRC[[78]](#footnote-78) معرفی شد و امروزه به عنوان استاندارد مقایسه عملکرد شبکه‌های عصبی پیچشی می‌باشد. امروزه این مجموعه‌داده بیشتر از 14 میلیون تصویر در غالب بیش از 20 هزار طبقه دارد. مدل‌های جدید طراحی شده با استفاده از این مجموعه‌داده آموزش می‌بینند و اعتبارسنجی ‌می‌شوند.

در صورتی که مجموعه‌داده‌ی کوچکی با مثلا 20 هزار تصویر موجود باشد، استفاده از روش آموزش از ابتدا مناسب نیست، چرا که هم تعداد تصاویر ممکن است برای آموزش ناکافی باشد و هم لازم است برای همگرا شدن وزن‌های مدل، تعداد دوره‌های آموزشی بسیار بیشتری طی شود. برای حل این مشکلات از تکنیک یادگیری انتقالی استفاده می‌شود. در یادگیری انتقالی، از معماری شبکه‌های عصبی که پیش از آن بر روی مجموعه‌داده‌های عظیم دیگر مانند ImageNet آموزش دیده شده اند و وزن‌های لایه‌های آنها به اعداد مناسب همگرا شده است استفاده می‌شود. این مدل‌ها را بر روی مجموعه‌داده مورد نظر آموزش می‌دهند و وزن‌های لایه‌های آنها به جای تغییرات بزرگ، تغییرات کمی می‌کنند و اصطلاحا تنظیم دقیق[[79]](#footnote-79) می‌شوند. استفاده از این نوع یادگیری به دلیل عدم استفاده از وزن‌گذاری تصادفی و همگرا بودن وزن لایه‌ها در ابتدا، هم سریعتر است و هم دقت بالاتری در فرایند ارزیابی مجموعه‌داده جدید ارائه می‌دهد [7, 23]. در جدول 3-2، مشخصات مدل‌های استفاده شده در این تحقیق نمایش داده شده است.

|  |
| --- |
| جدول ‏3-2- مقایسه بین مدل‌های از پیش‌آموزش دیده‌شده مورد استفاده در این تحقیق |
| |  |  |  |  |  | | --- | --- | --- | --- | --- | | دقت ImageNetV1 | قدرت مورد نیاز | تعداد پارامتر‌ها | | نام مدل | | 84.22 | 8.37 گیگا فلاپ | 21.5 میلیون | **EfficientnetV2S** | | | 85.11 | 24.58 گیگا فلاپ | 54.1 میلیون | **EfficientnetV2M** | | | 76.13 | 4.09 گیگا فلاپ | 25.6 میلیون | **Resnet50** | | | 77.61 | 4.23 گیگا فلاپ | 25.0 میلیون | **Resnext5032x4d** | | | 78.46 | 11.4 گیگا فلاپ | 68.9 میلیون | **WideResnet50** | | | 80.03 | 8.47 گیگا فلاپ | 39.4 میلیون | **RegnetY8GF** | | | 82.02 | 5.94 گیگا فلاپ | 28.4 میلیون | **SwinV2T** | | | 83.71 | 11.55 گیگا فلاپ | 49.7 میلیون | **SwinV2S** | | | 82.52 | 4.46 گیگا فلاپ | 28.6 میلیون | **ConvNextT** | | | 83.61 | 8.68 گیگا فلاپ | 50.2 میلیون | **ConvNextS** | | |

## الگوریتم‌های بهنیه سازی ازدحامی

الگوریتم‌های بهینه سازی ازدحامی[[80]](#footnote-80) ، خانواده‌ای از الگوریتم‌های بهینه‌سازی هستند که رفتار موجودات در طبیعت الهام گرفته شدند. این الهام ممکن است از رفتارهای سازمان‌یافته در موجوداتی مانند ازدحام پرندگان، ماهی‌ها، کلونی‌های مورچه‌ها و باقی موجودات اجتماعی باشد. در سال‌های اخیر به این الگوریتم‌ها در زمینه هوش محاسباتی[[81]](#footnote-81) به خاطر توانایی آنها در حل مسائل بهینه‌سازی پیچیده، حتی در ابعاد بالا و فضای غیر خطی، توجه فراوانی شده‌است. ریشه‌ی این الگوریتم‌ها در اصول هوش ازدحامی[[82]](#footnote-82) موجودات نهفته است، که شامل شبیه‌سازی حرکت، مسیریابی، رفتار‌های اجتماعی و شکار توسط جاندارن را انجام می‌دهند. به عنوان مثال، بهینه‌سازی کلونی مورچه‌ها از جمع‌آوری آذوقه توسط مورچه‌ها، بهینه‌سازی ازدحام ذرات از حرکت دسته‌جمعی پرندگان و الگوریتم ژنتیک از روند انتخاب طبیعی در ژنتیک الهام گرفته شده اند. هوش ازدحامی بخشی از هوش مصنوعی می‌باشد که به بررسی این رفتارهای طبیعی در حیوانات می‌پردازد. ایده‌ی استفاده شده در این بخش، این است که رفتار‌های تکی و جداگانه هر نماینده[[83]](#footnote-83) در کنار ارتباط گرفتن آنها با یکدیگر برای رسیدن به هدفشان، می‌تواند باعث ظهور رفتارهایی شود که از خود هوش و سازگاری با محیط نشان می‌دهند. ترکیب الهام از طبیعت و محاسبات رایانه‌ای، الگوریتم‌های ازدحامی را به ابزاری قدرتمند برای پیشرفت حل مسائل پیچیده در دنیای واقعی تبدیل کرده است. عناصر مهم تشکیل دهنده این الگوریتم‌ها معمولا شامل نماینده‌ها، تابع هدف، ساختار همسایگی[[84]](#footnote-84) و استراتژی‌های حرکتی[[85]](#footnote-85) است. بنابراین، الگوریتم‌های بهینه‌سازی بر پایه ازدحام، راه حل مناسبی برای حل مسائل پیچیده و دارای فضای غیر خطی می‌باشند. توانایی آنها در شبیه‌سازی رفتارهای اجتماعی و تعمیم دادن آنها به ابعاد مسائل پیچیده، باعث پیشرفت‌های فراوان در هوش محاسباتی شده است. محققین همواره به دنبال ابداع روش‌های بهینه‌سازی ازدحامی جدید و افزایش کاربردهای آن در زمینه‌های مهندسی، علوم و تصمیم‌گیری هستند. پیش‌بینی می‌شود تاثیر این الگوریتم‌ها در سال‌های آینده بسیار بیشتر شود.

## نرم‌افزار و سخت‌افزارهای لازم برای آموزش شبکه عصبی

### CPU

CPU پردازنده مرکزی هر رایانه‌ای می‌باشد. در کنار حافظه تصادفی[[86]](#footnote-86) ، بسیاری از محاسبات رایانه در این بخش‌ها اجرا می‌شود. این بخش تمام ورودی‌ها و خروجی‌ها را کنترل کرده و به عنوان مغز رایانه می‌شود. چند تراشه کوچک در درون CPU وجود دارند که وظیفه انجام محاسبات را دارند. هر تراشه از میلیاردها ترانزیستور تشکیل شده است که سرعت پردازش CPU را تعیین می‌کنند. امروزه سرعت پردازش این واحد بین 1 تا 10 گیگاهرتز می‌رسد.

### GPU

واحد پردازش گرافیکی یا GPU، بخشی از رایانه است که توانایی پردازش اطلاعات تصویری را دارد. علاوه بر این، تفاوت مهم دیگر آن با CPU، توانایی پردازش اطلاعات به صورت موازی است. در CPU اطلاعات به صورت سری و پشت سر هم پردازش می‌شوند اما با سرعت بسیار بالا. در GPU اطلاعات با سرعت کنتری پردازش می‌شوند، اما این امکان وجود دارد که چندین محاسبه به صورت همزمان و برروی حافظه مخصوص آن پردازش شوند. این امر، استفاده از GPU‌ها را در زمینه هوش مصنوعی و خصوصا طبقه‌بندی تصاویر مهم می‌سازد. برای بارگذاری مدل‌های شبکه عصبی و تصاویر بر روی حافظه‌های GPU، ابتدا لازم است آنها را به ساختار خاصی از آرایه‌ها به نام تنسور[[87]](#footnote-87) تبدیل شوند.

### پایتون

پایتون یا (Python) یک زبان برنامه نویسی سطح بالا بوده و در بسیاری از زمینه‌ها استفاده می‌شود. این زبان اولین بار در سال 1980 توسط گودیو[[88]](#footnote-88) ابداع شد. پس از گذشت سال‌ها، از سال 2010 تا به امروز به عنوان یکی از محبوب‌ترین زبآنهای برنامه نویسی شناخته شده است. تا به امروز کتابخانه‌های پرکابرد و فراوان برای آن ساخته شده است و به صورت رایگان قابل استفاده هستند. امروزه زبان پایتون در زمینه علوم داده و هوش مصنوعی محبوب‌ترین زبان برنامه نویسی است و از آن برای طراحی سیستم‌های قابل توسعه و دارای شبکه‌ی عصبی استفاده می‌شود.

## شاخص عملکرد

در هوش مصنوعی و یادگیری تحت نظارت، یک مجموعه‌داده به مدل داده می‌شود و مدل آموزش می‌بیند. سپس مدل باید بتواند بر روی مجموعه‌داده‌ای که تا به حال مشاهده نکرده است، داده‌ها را طبقه بندی کند. برای بررسی عملکرد طبقه‌بندی این مدل، باید از روش‌های استانداردی برای مقایسه آن با مدل‌های دیگر استفاده شود. اگر داده طبقه‌بندی شده به دسته مورد نظر تعلق داشته باشد، نتیجه را مثبت گرفته و در صورتی که به دسته مورد نظر تعلق نداشته باشد، آن را منفی در نظر میگیریم. 4 حالت اصلی برای هر داده خواهیم داشت:

* درصورتی که داده طبقه‌بندی شده مثبت باشد و طبقه‌ای که باید در آن در آن قرار داشته باشد مثبت باشد، مثبت صحیح یا [[89]](#footnote-89)TP حساب می‌شود.
* درصورتی که داده طبقه‌بندی شده منفی باشد و طبقه‌ای که باید در آن در آن قرار داشته باشد منفی باشد، منفی صحیح یا TN[[90]](#footnote-90) حساب می‌شود.
* اگر داده طبقه‌بندی شده مثبت باشد اما طبقه‌ای که باید در آن قرار داشته باشد منفی باشد، مثبت غلط یا FP[[91]](#footnote-91) حساب می‌شود.
* اگر داده طبقه‌بندی شده منفی باشد اما طبقه‌ای که باید در آن قرار داشته باشد مثبت باشد، منفی غلط یا FN[[92]](#footnote-92) حساب خواهد شد.

بر اساس چهار حالت اصلی ذکر شده در مسائل طبقه‌بندی، چند شاخص مهم برای سنجش توانایی مدل‌های شبکه‌ی عصبی در طبقه‌بندی ارائه شده است:

* شاخص دقت: این شاخص اساسی‌ترین و مهمترین معیار برای سنجش توانایی طبقه‌بندی است.

|  |  |
| --- | --- |
| **(رابطه 3-6)** |  |

* شاخص یادآوری: این شاخص برابر است با تقسيم تعداد مواردی كه توسط مدل درست تشخيص داده‌شده بر تعداد كل مواردی كه توسط مدل پيش‌بيني شده‌اند.

|  |  |
| --- | --- |
| **(رابطه 3-7)** |  |

* شاخص صحت: این شاخص نشان‌دهنده عملکرد مدل در درست تشخیص دادن مثبت‌ها است.

|  |  |
| --- | --- |
| **(رابطه 3-8)** |  |

* امتیاز اف-1: این شاخص ترکیبی از دو شاخص قبلی است و یک نوع میانگین بین آنها می‌باشد.

|  |  |
| --- | --- |
| **(رابطه 3-9)** |  |

# فصل چهارم

آموزش شبکه و نتایج

## مقدمه

روش‌های سنتی شناسائی بیماری‌های گیاهی و تجویز داروی مناسب برای آنها به تعداد قابل توجهی از افراد متخصص نیازمند است و سرعت آن پایین می‌باشد. اما با توسعه و پیشرفت شبکه‌های عصبی پیچشی و توانایی حل مسائل پردازش تصویر پیچیده توسط آنها، این امکان فراهم شده است که بتوان شبکه‌ی عصبی مناسبی برای شناسائی بیماری برگ گیاهان و در صورت امکان تجویز داروی مناسب آن را طراحی و ارائه کرد. به منظور دستیابی به این هدف، تعدادی از مدل‌های شبکه‌های عصبی پیچشی از پیش آموزش دیده شده برروی مجموعه‌داده‌ی چالش Plant Pathology 2021 [27] به صورت چند برچسبه و با استفاده از تکنیک یادگیری انتقالی آموزش می‌بینند و در نهایت نتایج به دست آمده از این مدل‌ها با یکدیگر مقایسه می‌شوند. در آزمون دیگری، با استفاده از الگوریتم‌‌های بهینه‌سازی ازدحامی، حدود آستانه‌های هر برچسب تغییر یافته و دقت مدل افزایش می‌یابد.

## مجموعه‌داده

در این تحقیق از مجموعه‌داده‌ی مربوط به هشتمین چالش طبقه‌بندی بصری دقیق که در سال 2021 برگزار شد، استفاده شده است. این مجموعه‌داده شامل 18633 تصویر در قالب 12 طبقه اصلی می‌باشد. در زیر نمونه‌ای از تصاویر آمده است.

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
|  | | | | | |
| الف) لکه چشم غوربافه‌ای | ب) زخم | ج) کپک پودری | د) زنگار | ه) زنگار | و) نامشخص |
| شکل 64- نمونه‌ای از بیماری‌های موجود در مجموعه‌داده مورد استفاده در این تحقیق: الف) لکه چشم غورباقه‌ای (frog eye leaf spot) ب) زخم سیم (scab) ج) کپک پودری (powdery mildew) د) زنگار (rust) ه) زنگار و و) نامشخص (complex) [27] | | | | | |

از میان 12 طبقه موجود، یکی از آنها مربوط به تصاویر برگ‌های سالم می‌باشد. 5 تا از این طبقه‌ها که هر کدام شامل یک عامل بیماری هستند در شکل 1-4 نمایش داده شده است. 6 طبقه دیگر شامل ترکیبی از این 5 بیماری است، که شامل موارد زیر می‌باشد:

* Scab / Frog eye leaf spot
* Scab / Frog eye leaf spot / Complex
* Frog eye leaf spot / Complex
* Rust / Frog eye leaf spot
* Rust / Complex
* Powdery Mildew / Complex

|  |
| --- |
|  |
| شکل 65- نمودار فراوانی مربوط به هر طبقه موجود در مجموعه‌داده Plant Pathology 2021 |
|  |
| شکل 66- نمودار فراوانی درصدی مربوط به مجموعه‌داده Plant Pathology 2021 |

شکل 4-2 و 4-3 نمودارهای فراوانی داده‌ها را نشان می‌دهد. همانطور که مشاهده می‌شود، داده‌های موجود در این مجموعه‌داده بسیار نامتوازن است. بنا براین باید از تکنیک‌های مناسبی در آموزش مدل‌ها استفاده نمود. یک از اهداف این تحقیق، طراحی شبکه‌ی عصبی پیچشی برای شناسائی یک یا چند بیماری موجود بر روی برگ‌های گیاه می‌باشد. برای دستیابی به این هدف و همچنین جبران بخشی از مشکل نامتوازن بودن داده‌های هر طبقه، در این تحقیق از تبدیل طبقات به مجموعه برچسب‌ها استفاده شده است. در این روش، طبقه‌هایی که شامل چند برچسب هستند (مانند Rust / Complex)، برچسب‌هایشان به صورت جداگانه در نظر گرفته می‌شوند. یعنی هر عکس می‌تواند یک یا چند برچسب داشته باشد و در چند تا از طبقه‌های اصلی قرار بگیرد. به طور مثال، اگر فرض شود در حالت اول تصویری به طبقه [Scab / Frog eye leaf spot / Complex] تعلق دارد، حالا با استفاده از این روش به سه طبقه‌ی Scab و Frog eye leaf spot و Complex تعلق دارد و هر سه برچسب را دارد. اگر در حالت اول به مسئله نگاه شود، مسئله از نوع طبقه‌بندی چند طبقه است و اگر حالت دوم در نظر گرفته شود، مسئله از نوع طبقه‌بندی چند برچسبه حساب می‌شود. در حالت اول، خروجی شبکه عصبی باید به صورت عددی بین 0 تا 11 (برابر با تعداد طبقه‌ها و در اینجا 7) باشد و در حالت دوم، این خروجی باید به صورت آرایه‌ای با 6 خانه (برابر با تعداد طبقه‌های تک برچسبه و در اینجا [0و1و0و1و0و1]) باشد. در صورت استفاده از این تکنیک، نمودار فراوانی داده‌ها به شکل 4-4 تبدیل می‌شود:

|  |  |
| --- | --- |
|  |  |
| (الف) | (ب) |
| شکل 67- نمودار فراوانی داده‌های Plant Pathology 2021 به صورت چندبرچسبه الف) بر اساس تعداد ب) بر اساس درصد داده‌‎ها | |

با استفاده از تکنیک ذکرشده، این امکان به وجود می‌آید که ترکیب‌های دیگری از بیماری‌ها که در طبقه‌های مجموعه‌داده موجود نمی‌باشد نیز شناسائی شوند. چرا که شبکه آموزش می‌بیند تا برچسب‌ها را به صورت مجزا از هم یاد بگیرد و در صورت لزوم، ترکیبی از برچسب‌ها که در طبقه‌های اولیه موجود نمی‌باشد شناسائی شوند. علاوه بر آن، در صورت نگاه کردن به مسئله به صورت چند برچسبه، بخشی از مشکل نامتوازن بودن داده‌ها برطرف می‌شود. همچنین در این تحقیق، از تکنیک‌های داده‌افزایی جهت افزایش تعداد تصاویر موجود برای رفع مشکل عدم توازن داده‌ها استفاده شده است. این تکنیک‌ها شامل انعکاس آیینه‌ای، چرخش، تغییر مرکز تصویر، تغییرات روشنایی تصاویر و تغییر تار و یا واضح بودن تصاویر به صورت تصادفی استفاده شده است. جدول 4-1 و 4-2 فراوانی داده‌ها را نشان می‌دهند.

جدول ‏4-1- فراوانی‌داده‌های مجموعه‌داده Plant Pathology 2021 به صورت چند طبقه.

|  |  |  |
| --- | --- | --- |
| درصد سهم | تعداد داده‌ها | نام طبقه |
| 25. 9 | 4826 | **Scab** |
| 24. 8 | 4624 | **Healthy** |
| 17. 1 | 3181 | **Frog\_eye\_leaf\_spot** |
| 10. 0 | 1860 | **Rust** |
| 8. 6 | 1602 | **Complex** |
| 6. 4 | 1184 | **Powdery\_mildew** |
| 3. 7 | 689 | **Scab, Frog\_eye\_leaf\_spot** |
| 1. 1 | 200 | **Scab, Frog\_eye\_leaf\_spot, Complex** |
| 0. 9 | 165 | **Frog\_eye\_leaf\_spot, Complex** |
| 0. 7 | 122 | **Rust, Frog\_eye\_leaf\_spot** |
| 0. 5 | 97 | **Rust, Complex** |
| 0. 5 | 87 | **Powdery\_mildew, Complex** |

جدول ‏4-2- فراوانی داده‌های مجموعه‌داده Plant Pathology 2021 پس از عملیات تبدیل طبقات اصلی به حالت چند برچسبه

|  |  |  |  |
| --- | --- | --- | --- |
| تعداد داده‌هایی که ندارند | درصد سهم داده‌هایی که دارند | تعداد داده‌هایی که دارند | نام برچسب |
| 12920 | 28. 3 | 5712 | **Scab** |
| 14008 | 22. 9 | 4624 | **Healthy** |
| 14280 | 21. 6 | 4352 | **Frog\_eye\_leaf\_spot** |
| 16481 | 10. 7 | 2151 | **Complex** |
| 16555 | 10. 3 | 2077 | **Rust** |
| 17361 | 6. 3 | 1271 | **Powdery\_mildew** |

## توسعه و روند آموزش مدل شبکه عصبی

در این قسمت به فرایند آموزش شبکه‌های عصبی پیچشی استفاده شده در این تحقیق می‌پردازیم. در مسائل طبقه‌بندی تصاویر با استفاده از یادگیری عمیق، هر تصویر به شبکه‌ی عصبی داده می‌شود و شبکه یک خروجی تولید می‌کند. با استفاده از اختلاف خروجی شبکه و مقداری که باید تولید شود که همان برچسب‌های موجود در مجموعه‌داده هستند، شبکه خود را اصلاح می‌کند تا دفعه بعد، بهتر بتواند تصویری را طبقه‌بندی کند. در این تحقیق از تکنیک‌های مختلفی برای آموزش بهتر و افزایش توانایی طبقه‌بندی شبکه استفاده شده است که به آنها می‌پردازیم.

### مشخصات سخت‌افزار و نرم‌افزار جهت پیاده‌سازی شبکه

برای پیاده‌سازی آموزش شبکه‌های عصبی پیچشی موجود در این تحقیق، از زبان برنامه‌نویسی پایتون نسخه 3 استفاده شده است. این نرم افزار در محیط Jupyter Notebook اجرا شده است و سخت افزار مورد استفاده نیز یک رایانه خانگی شامل پردازنده I7-7700 و در کنار 16 گیگابایت حافظه تصادفی و یک پردازنده گرافیکی GTX 1080Ti شامل 11 گیگابایت حافظه گرافیکی[[93]](#footnote-93) است. محیط برنامه‌نویسی و همچنین داده‌ها بر روی یک حافظه جامد[[94]](#footnote-94) قرار دارند و از آن جا پیاده‌سازی می‌شوند.

### پلتفرم Pytorch

پایتورچ یک کتابخانه‌ی متن‌باز[[95]](#footnote-95) است که در زمینه‌ی یادگیری عمیق استفاده می‌شود. در سال‌های اخیر شرکت Meta این پلتفرم را برای عملیات‌های یادگیری عمیق و پردازش تصویر خود ایجاد کرد و همچنین در دسترس عموم قرار داد. استفاده از این پلتفرم امکان پیاده‌سازی شبکه‌های عصبی پیچشی به صورت ساده به کاربر داده و همچنین به راحتی می‌توان فرایند‌های ساخت و آموزش شبکه را مدیریت و دستکاری کرد. رقیب پایتورچ، پلتفرم Tensorflow می‌باشد که چندین سال قبل از پایتورچ منتشر شده بود و امکانات پیاده‌سازی و آموزش شبکه را دارد. در صورت مقایسه این دو پلتفرم، این نتیجه حاصل می‌شود که استفاده از پلتفرم تنسورفلو ساده‌تر است و زمان و خطوط کد کمتری برای راه‌اندازی و آموزش شبکه نیاز دارد. از طرفی، راه‌اندازی اولیه این پلتفرم بر روی یک رایانه کار دشوار‌تری می‌باشد. در مقابل، پایتورچ برای ایجاد و آموزش شبکه به تعداد خطوط کد بیشتری نیاز دارد ولی امکان تغییرات و دست‌کاری بیشتری به کاربر داده می‌شود. از آن جایی که تنسورفلو قدیمی‌تر است، پشتیبانی، وبسایت‌های بحث و تبادل نظر و مدل‌های از پیش آموزش دیده‌شده بیشتری دارد. در مقابل در سال‌های اخیر، پایتورچ پیشرفت‌های بسیار زیادی کرده و امکانات جدیدتری به آن اضافه شده است. کاربران تمایل بیشتری برای استفاده از پایتورچ از خود نشان‌داده اند و تعداد کاربران آن در سال‌های اخیر بیشتر شده است. در مقابل، تعداد استفاده‌کنندگان از پلتفرم تنسورفلو کمتر شده است. به علاوه انتخاب محبوب و مورد پسند محققان برای انجام پژوهش‌هایشان، پایتورچ می‌باشد. در این تحقیق نیز برای پیاده‌سازی شبکه‌های استفاده‌شده، از پلتفرم پایتورچ استفاده شده است.

### تقسیم بندی داده‌ها

در بخش 4-2 به بررسی مجموعه‌داده پرداخته شد. با توجه به تعداد داد‌ه‌ها و ساختار مجموعه‌داده، از روش چندبرچسبه سازی و برخی تکنیک‌های داده‌افزایی استفاده شده است. تصاویر موجود در این مجموعه‌داده، ابتدا به صورت تصادفی و با حفظ نسبت داده‌های هر 12 طبقه‌ی اولیه، به نسبت 80%-20% جهت آموزش و آزمون جداسازی می‌شوند. از این 20% برای سنجش نهایی مدل و از 80% باقی‌مانده جهت آموزش مدل‌ها با استفاده از تکنیک یادگیری انتقالی استفاده می‌شود.

### تقسیم‌بندی داد‌های آموزشی

در این تحقیق از تکنیک ارزیابی 5-قسمتی برای تقسیم‌بندی داده‌های آموزشی استفاده شده. در این تکنیک، ابتدا داده‌های آموزشی در 5 دسته‌ی مساوی با رعایت برابر بودن داده‌های هر دسته تقسیم می‌شوند. سپس مدل‌های شبکه‌های عصبی در 5 حالت مختلف، هر بار با در نظر گرفتن یکی از دسته‌ها برای ارزیابی یا Validation، با استفاده از 4 دسته دیگر مدل‌ها آموزش دیده می‌شوند. پس از انجام این پنج حالت، بهترین مدل با عملکرد بالاتر انتخاب شده و برای تست نهایی بر روری داده‌های تست استفاده می‌شود. شکل 4-5 این روش را نمایش می‌دهد.

|  |
| --- |
|  |
| شکل 68-فرایند آموزش با استفاده از تکنیک ارزیابی 5-قسمتی. رنگ سبز نشان‌دهنده دسته‌هایی است که برای فرایند آموزش استفاده می‌شوند و رنگ نارنجی نشان‌دهنده دسته‌هایی است که برای ارزیابی مدل‌ها در حین آموزش استفاده می‌شوند |

### مدل‌های شبکه‌ی عصبی پیچشی

همانطور که اشاره شده بود، مجموعه‌داده‌ی استفاده شده در این تحقیق کوچک می‌باشد و در صورت آموزش شبکه به صورت خام و از ابتدا، وزن‌های اولیه مدل‌ها به صورت تصادفی انتخاب می‌شوند و فرایند آموزش مدل‌ها برای همگرا‌شدن وزن‌ها نیاز به تعداد دوره‌های آموزشی بسیار بیشتری خواهند داشت. روش کارآمدتری که در اکثر تحقیقات مرتبط با زمینه پردازش تصویر در کشاورزی نیز استفاده می‌شود، استفاده از تکنیک یادگیری انتقالی می‌باشد. در این تکنیک، مدل‌های شبکه‌ی عصبی پیچشی از پیش آموزش دیده توسط پلتفرم پایتورچ ابتدا بارگذاری می‌شوند. نوع یادگیری انتقالی مورد استفاده در اینجا، آموزش کل لایه‌های شبکه و تنظیم دقیق آنها در کنار ابرپارامترها می‌باشد. پس از آن، در لایه‌ی اول هر شبکه که وظیفه استخراج ویژگی‌های اولیه را دارند، لایه بلوک برون‌انداز قرار داده می‌شود. همچنین لایه تمام‌متصل تعویض شده و به ترتیب یک لایه برون انداز، لایه تمام‌متصل با تعداد ورودی ویژگی‌ها و خروجی 6 قرار داده شده و در نهایت نیز تابع فعالسازی سیگموید افزوده می‌شود. وضوح تصاویر ورودی در تمامی شبکه‌های مورد آزمون، برابر با 400 پیکسل در 400 پیکسل می‌باشد.

### نرم‌سازی برچسب‌ها

تکنیک نرم‌سازی برچسب‌ها[[96]](#footnote-96) در سال 2015 توسط محققین گوگل ابداع شد [86]. در این تکنیک برچسب‌های مربوط به هر داده به جای منطق مطلق 0 و 1 (و یا اعداد حسابی بین 0 و . . . )، به صورت عددی اعشاری بین 0 و 1 ( و یا اعداد حسابی بین 0 و . . . ) در می‌آیند. رابطه آن به صورت زیر می‌باشد که در آن K نشان‌دهنده تعداد طبقات موجود، ε نرخ نرم‌سازی، δ عدد مربوط به عدد برچسب یک داده و q’ نشان دهنده عدد نهایی نسبت داده‌ شده آن برچسب به داده مورد نظر می‌باشد:

|  |  |
| --- | --- |
| **(رابطه 4-1)** | **q’(k) = (1 – ε)δk,y +** [86] |

به طور مثال، اگر فرض شود که داده‌ای برچسب [1.0.1.0.0.0] را داشته باشد و شامل دو برچسب باشد. حال اگر نرخ نرم‌سازی 0.2 باشد، از آن جایی که تعداد طبقات 6 تا می‌باشد، برچسب‌های این داده به صورت [0.83,0.03,0.83,0.03,0.03,0.03] تبدیل می‌شود. از مزایای استفاده از این روش می‌توان به افزایش عملکرد مدل‌ها و فشرده‌شدن بیشتر داده‌های مشابه در یک نقطه می‌باشد. از آن جایی که گمان می‌رود تصاویر موجود در مجموعه‌داده مورد استفاده حاوی نویز فراوان باشد (به طور مثال برچسب Complex)، در این تحقیق از نرم‌سازی برچسب‌ها استفاده شده و نرخ آن برابر با 0.2 است. شکل 4-6 مقایسه‌ای بین تاثیر نرخ‌های نرم‌سازی بر برچسب‌ها انجام می‌دهد.

|  |  |  |
| --- | --- | --- |
|  |  |  |
| الف) نرخ نرم‌سازی=0.0 | ب) نرخ نرم‌سازی=0.2 | ج) نرخ نرم‌سازی=0.7 |
| شکل 69- مقایسه تاثیر نرخ‌های مختلف نرم‌سازی برچسب‌ها بر روی داده‌های مجموعه‌داده Cifar-10 الف) نرخ نرم‌سازی صفر ب) نرخ نرم‌سازی 0.2 ج) نرخ نرم‌سازی 0.7 [115] | | |

### تابع هزینه

در فرایند آموزش تحت نظارت شبکه‌ی عصبی عمیق، پس از هربار پیش‌بینی شبکه از داده‌های ورودی و نسبت‌دادن خروجی آن، اختلافی بین خروجی و نتیجه‌ای که باید داشته باشد به وجود می‌آید. برای محاسبه این اختلاف، روش‌ها و توابع مختلفی وجود دارد. در صورتی که نوع طبقه‌بندی، طبقه‌بندی چندطبقه (Multi Class) باشد، توابع هزینه مختلفی وجود دارد مانند Categorical Cross Entropy و Cross Entropy. اما در زمینه طبقه‌بندی چندبرچسبه تعداد بسیار کمتری از توابع هزینه وجود دارند. از آن جایی که از پلتفرم پایتورچ در این تحقیق استفاده شده است، تابع هزینه مورد استفاده در آن، Binary Cross Entropy می‌باشد. رابطه 4-2 مربوط به این تابع هزینه می‌باشد. در این رابطه، yi برچسب i ام برای داده مورد بررسی و احتمال واقعی 1 بودن آن برچسب است:

|  |  |
| --- | --- |
| **(رابطه 4-2)** |  |

### الگوریتم بهینه‌سازی

پس از محاسبه هزینه توسط تابع هزینه، نوبت به انجام الگوریتم پس‌انتشار و حرکت به سمت عقب و تغییر وزن‌های لایه‌های شبکه عصبی می‌رسد. وظیفه انجام این کار بر عهده الگوریتم بهینه‌سازی است. برای بهینه‌سازی شبکه‌های عصبی پیچشی در این تحقیق، از الگوریتم Adam استفاده شده است. این بهینه‌ساز، الگوریتم گرادیان کاهشی را سرعت می‌بخشد و مهم‌ترین ویژگی‌های AdaGrad و RMSProp را استفاده می‌کند تا بتواند الگوریتم بهینه‌سازی مناسبی برای گرادیآنهای پخش‌شده برروی برچسب‌های نویز دار ارائه دهد [80].

### نرخ یادگیری

نرخ یادگیری از مهم‌ترین ابرپارامترهای آموزش شبکه در یادگیری عمیق است. نرخ یادگیری می‌تواند بین 0 و 1 باشد. به هنگام عملیات پس‌انتشار، بعد از محاسبه‌ی هزینه مربوط به هر لایه، میزان تغییرات در نرخ یادگیری ضرب می‌شود تا به طور کامل و 100% برروی وزن‌ها اعمال نشود، چرا که در این صورت شبکه به سرعت دچار مشکل بیش‌برآزش می‌شود. نرخ یادگیری به طور مستقیم توسط الگوریتم بهینه‌سازی اعمال می‌شود. در این تحقیق از یک تابع تغییردهنده نرخ یادگیری استفاده شده که یه صورت کاهشی و کسینوسی عمل می‌کند. این الگوریتم Cosine Annealing LR نام دارد که از نرخ یادگیری 0.0001 شروع کرده و تا 0.00001 کاهش پیدا می‌کند و در دوره‌های آخر روی همان عدد ثابت می‌ماند.

### سایر ابرپارامتر‌های آموزش شبگه

پس از انجام آزمایش‌های اولیه و با توجه به نتایج به دست آمده در [7, 23, 75, 76]، مشخص شده است که تعداد دوره‌های آموزش مناسب برای این این تحقیق، عدد 30 می‌باشد. با توجه به حجم مختلف شبکه‌های پیچشی و حجم محدود حافظه گرافیکی، داده‌ها با سایز‌های 12 و 20 دسته بندی‌می‌شوند و به مدل‌ها داده می‌شوند. از آن جایی که نوع طبقه‌بندی به کار رفته در این تحقیق چند برچسبه است، باید آستانه‌ای برای مشخص کردن پیش‌بینی‌های شبکه و محاسبه دقت آن ارائه شود. پس از انجام آزمایش‌های متعدد، مشخص شد که حد آستانه 0. 25 برای این کار مناسب‌تر است و در صورتی که خروجی هر برچسب از یک شبکه از این عدد بیشتر باشد، یعنی تصویر ورودی این برچسب را دارا می‌باشد. ‌مدل‌های شبکه عصبی پس از آموزش ذخیره ‌شده و با استفاده از داده‌های بخش تست، مورد ارزیابی قرار گرفته می‌شوند.

## آزمون‌ها و نتایج

این بخش به ارائه نتایج و اعداد به‌دست آمده از آموزش و ارزیابی مدل‌های شبکه عصبی پیچشی آموزش دیده‌شده می‌پردازد. فرایند آموزش به این صورت می‌باشد که در ابتدا، مدل‌ شبکه‌عصبی از پیش آموزش دیده شده فراخوانی می‌شود، در لایه‌های آن تغییر ایجاد شده و لایه‌های برون‌انداز، بلوک برون‌انداز و تابع سیگموید به آن اضافه می‌شوند. همچنین لایه‌ی تمام‌متصل جدیدی برای طبقه‌بندی در بین 6 برچسب تغییر داده می‌شود. سپس از بین مجموعه‌ی داده‌های آموزشی، چهار قسمت برای آموزش مدل و یک قسمت برای اعتبارسنجی در حین آموزش و جلوگیری از فرایند بیش‌برآزش استفاده می‌شود. در 5 حالت متفاوت که هر بار یکی از این 5 قسمت برای اعتبارسنجی استفاده شده، مدل‌ها آموزش دیده می‌شوند و در نهایت مدلی که بیشترین دقت را دارد، برای ارزیابی نهائی بر روی داده‌های آزمون انتخاب می‌شوند. برای پیش‌بینی آن که تصویری برچسبی دارد یا نه، باید خروجی مدل برای آن برچسب بزرگتر از حد آستانه تعیین شده باشد (در اینجا 0.25). از این روش برای سنجش دقت مدل‌ها استفاده شده است. بر روی تصاویر مورد آموزش، تغییرات داده‌افزایی چون چرخش، تغییر مرکز تصاویر، انعکاس آیینه‌ای، تغییر روشنایی و تغییر واضح سازی تصاویر به صورت تصادفی اعمال می‌شود و تعداد دوره‌های آموزشی 30 بار است. سایز دسته‌های تصاویر با توجه به بزرگ بودن یا نبودن مدل‌ها 10 و یا 20 می‌باشد. در آخر فرآیند آموزش نیز، مدل‌ها ذخیره شده و از آن‌ها برای ارزیابی بر روی داده‌های آزمون استفاده می‌شود. در مرحله آخر، از تعدادی الگوریتم بهینه‌سازی بر پایه ازدحام جهت بهینه‌سازی حدود آستانه هرکدام از برچسب‌ها استفاده می‌شود و ابن نتایج بر روی داده‌های آزمون، ارزیابی می‌شوند.

### آزمون اول، آموزش و ارزیابی شبکه‌های عصبی پیچشی

در آزمون اول، تمام شبکه‌ها با استفاده از تکنیک ارزیابی چند قسمتی آموزش دیده می‌شوند و سپس بر روی داده‌های آزمون ارزیابی نهائی می‌شوند.

#### نتایج مدل EfficientnetV2S در آزمون اول

شکل 4-7، نمودارهای حاصل شده از فرایند آموزش شبکه EfficientnetV2S را نمایش می‌دهد. جدول 4-3، نتایج حاصل از ارزیابی این شبکه برروی داده‌های آزمون را نشان می‌دهد.

|  |  |
| --- | --- |
| جدول ‏4-3- درصد شاخص‌های عملکرد مدل EfficientnetV2S برای هر برچسب | |
| |  |  |  |  | | --- | --- | --- | --- | | شاخص امتیاز اف-1 | شاخص صحت | شاخص یادآوری | نام برچسب | | 94.56 | 94.56 | 94.56 | **Scab** | | 99.04 | 98.20 | 99.89 | **Healthy** | | 91.38 | 89.38 | 93.47 | **Frog eye leaf spot** | | 97.34 | 97.81 | 96.87 | **Rust** | | 85.58 | 82.01 | 89.73 | **Complex** | | 98.02 | 98.02 | 98.02 | **Powdery mildew** | | |
|  | |
|  |  |
| الف) نمودار دقت | ب) نمودار هزینه |
|  | |
| ج) نمودار امتیاز اف-1 | |
| شکل 70- نمودارهای حاصل شده در فرایند آموزش شبکه EfficientnetV2S الف) دقت ب) هزینه ج) امتیاز اف-1 | |

#### نتایج مدل EfficientnetV2M در آزمون اول

شکل 4-8، نمودارهای حاصل شده از فرایند آموزش شبکه EfficientnetV2M را نمایش می‌دهد. جدول 4-4، نتایج حاصل از ارزیابی این شبکه برروی داده‌های آزمون را نشان می‌دهد.

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| جدول ‏4-4- درصد شاخص‌های عملکرد مدل EfficientnetV2M برای هر برچسب   |  |  |  |  | | --- | --- | --- | --- | | شاخص امتیاز اف-1 | شاخص صحت | شاخص یادآوری | نام برچسب | | 93.95 | 93.18 | 94.74 | **Scab** | | 99.25 | 98.82 | 99.67 | **Healthy** | | 89.38 | 84.87 | 94.39 | **Frog eye leaf spot** | | 96.85 | 97.56 | 96.15 | **Rust** | | 84.43 | 80.92 | 88.27 | **Complex** | | 98.05 | 96.55 | 99.60 | **Powdery mildew** | | |
|  | |
|  |  |
| الف) نمودار دقت | ب) نمودار هزینه |
|  | |
| ج) نمودار امتیاز اف-1 | |
| شکل 71- نمودارهای حاصل شده در فرایند آموزش شبکه EfficientnetV2M الف) دقت ب) هزینه ج) امتیاز اف-1 | |

#### نتایج مدل Resnet50 در آزمون اول

شکل 4-9، نمودارهای حاصل شده از فرایند آموزش شبکه Resnet50 را نمایش می‌دهد. جدول 4-5، نتایج حاصل از ارزیابی این شبکه برروی داده‌های آزمون را نشان می‌دهد.

|  |  |
| --- | --- |
| جدول ‏4-5- درصد شاخص‌های عملکرد مدل Resnet50 برای هر برچسب | |
| |  |  |  |  | | --- | --- | --- | --- | | شاخص امتیاز اف-1 | شاخص صحت | شاخص یادآوری | نام برچسب | | 94.39 | 92.87 | 95.96 | **Scab** | | 99.09 | 98.30 | 99.89 | **Healthy** | | 91.97 | 90.29 | 93.70 | **Frog eye leaf spot** | | 97.81 | 98.77 | 96.87 | **Rust** | | 86.77 | 82.22 | 91.86 | **Complex** | | 98.00 | 98.79 | 97.23 | **Powdery mildew** | | |
|  | |
|  |  |
| الف) نمودار دقت | ب) نمودار هزینه |
|  | |
| ج) نمودار امتیاز اف-1 | |
| شکل 72- نمودارهای حاصل شده در فرایند آموزش شبکه Resnet50 الف) دقت ب) هزینه ج) امتیاز اف-1 | |

#### نتایج مدل Resnext5032x4D در آزمون اول

شکل 4-10، نمودارهای حاصل شده از فرایند آموزش شبکه Resnext5032x4D را نمایش می‌دهد. جدول 4-6، نتایج حاصل از ارزیابی این شبکه برروی داده‌های آزمون را نشان می‌دهد.

|  |  |
| --- | --- |
| جدول ‏4-6- درصد شاخص‌های عملکرد مدل Resnext5032x4D برای هر برچسب | |
| |  |  |  |  | | --- | --- | --- | --- | | شاخص امتیاز اف-1 | شاخص صحت | شاخص یادآوری | نام برچسب | | 95.37 | 95.03 | 95.70 | **Scab** | | 99.30 | 98.82 | 99.78 | **Healthy** | | 93.05 | 91.11 | 95.08 | **Frog eye leaf spot** | | 97.45 | 98.28 | 96.63 | **Rust** | | 88.73 | 86.23 | 91.38 | **Complex** | | 97.99 | 99.59 | 96.44 | **Powdery mildew** | | |
|  | |
|  |  |
| الف) نمودار دقت | ب) نمودار هزینه |
|  | |
| ج) نمودار امتیاز اف-1 | |
| شکل 73- نمودارهای حاصل شده در فرایند آموزش شبکه Resnext5032x4D الف) دقت ب) هزینه ج) امتیاز اف-1 | |

#### نتایج مدل WideResnet50 در آزمون اول

شکل 4-11، نمودارهای حاصل شده از فرایند آموزش شبکه WideResnet50 را نمایش می‌دهد. جدول 4-7، نتایج حاصل از ارزیابی این شبکه برروی داده‌های آزمون را نشان می‌دهد.

|  |  |
| --- | --- |
| جدول ‏4-7- درصد شاخص‌های عملکرد مدل Wideresnet50 برای هر برچسب | |
| |  |  |  |  | | --- | --- | --- | --- | | شاخص امتیاز اف-1 | شاخص صحت | شاخص یادآوری | نام برچسب | | 94.44 | 93.47 | 95.44 | **Scab** | | 99.46 | 99.04 | 99.89 | **Healthy** | | 91.42 | 89.57 | 93.36 | **Frog eye leaf spot** | | 97.96 | 97.38 | 98.55 | **Rust** | | 87.16 | 85.84 | 88.51 | **Complex** | | 98.23 | 97.28 | 99.20 | **Powdery mildew** | | |
|  | |
|  |  |
| الف) نمودار دقت | ب) نمودار هزینه |
|  | |
| ج) نمودار امتیاز اف-1 | |
| شکل 74- نمودارهای حاصل شده در فرایند آموزش شبکه Wideresnet50 الف) دقت ب) هزینه ج) امتیاز اف-1 | |

#### نتایج مدل RegnetY8GF در آزمون اول

شکل 4-12، نمودارهای حاصل شده از فرایند آموزش شبکه RegnetY8GF را نمایش می‌دهد. جدول 4-8، نتایج حاصل از ارزیابی این شبکه برروی داده‌های آزمون را نشان می‌دهد.

|  |  |
| --- | --- |
| جدول ‏4-8- درصد شاخص‌های عملکرد مدل RegnetY8GF برای هر برچسب | |
| |  |  |  |  | | --- | --- | --- | --- | | شاخص امتیاز اف-1 | شاخص صحت | شاخص یادآوری | نام برچسب | | 94.72 | 95.05 | 96.40 | **Scab** | | 98.25 | 96.67 | 100.00 | **Healthy** | | 88.80 | 92.04 | 88.67 | **Frog eye leaf spot** | | 96.75 | 96.86 | 96.39 | **Rust** | | 82.59 | 75.34 | 90.43 | **Complex** | | 96.35 | 98.75 | 94.07 | **Powdery mildew** | | |
|  | |
|  |  |
| الف) نمودار دقت | ب) نمودار هزینه |
|  | |
| ج) نمودار امتیاز اف-1 | |
| شکل 75- نمودارهای حاصل شده در فرایند آموزش شبکه RegnetY8GF الف) دقت ب) هزینه ج) امتیاز اف-1 | |

#### نتایج مدل SwinV2T در آزمون اول

شکل 4-13، نمودارهای حاصل شده از فرایند آموزش شبکه SwinV2T را نمایش می‌دهد. جدول 4-9، نتایج حاصل از ارزیابی این شبکه برروی داده‌های آزمون را نشان می‌دهد.

|  |  |
| --- | --- |
| جدول ‏4-9- درصد شاخص‌های عملکرد مدل SwinV2T برای هر برچسب | |
| |  |  |  |  | | --- | --- | --- | --- | | شاخص امتیاز اف-1 | شاخص صحت | شاخص یادآوری | نام برچسب | | 94.94 | 93.61 | 96.31 | **Scab** | | 99.46 | 98.93 | 100.00 | **Healthy** | | 91.42 | 89.04 | 93.93 | **Frog eye leaf spot** | | 97.59 | 97.82 | 97.35 | **Rust** | | 86.56 | 82.60 | 90.90 | **Complex** | | 98.60 | 99.20 | 98.02 | **Powdery mildew** | | |
|  | |
|  |  |
| الف) نمودار دقت | ب) نمودار هزینه |
|  | |
| ج) نمودار امتیاز اف-1 | |
| شکل 76- نمودارهای حاصل شده در فرایند آموزش شبکه SwinV2T الف) دقت ب) هزینه ج) امتیاز اف-1 | |

#### نتایج مدل SwinV2S در آزمون اول

شکل 4-14، نمودارهای حاصل شده از فرایند آموزش شبکه SwinV2S را نمایش می‌دهد. جدول 4-10، نتایج حاصل از ارزیابی این شبکه برروی داده‌های آزمون را نشان می‌دهد.

|  |  |
| --- | --- |
| جدول ‏4-10- درصد شاخص‌های عملکرد مدل SwinV2S برای هر برچسب | |
| |  |  |  |  | | --- | --- | --- | --- | | شاخص امتیاز اف-1 | شاخص صحت | شاخص یادآوری | نام برچسب | | 95.26 | 94.40 | 96.14 | **Scab** | | 99.41 | 98.83 | 100.00 | **Healthy** | | 91.88 | 89.92 | 93.93 | **Frog eye leaf spot** | | 97.47 | 97.59 | 97.35 | **Rust** | | 85.51 | 82.30 | 88.99 | **Complex** | | 99.00 | 99.20 | 98.81 | **Powdery mildew** | | |
|  | |
|  |  |
| الف) نمودار دقت | ب) نمودار هزینه |
|  | |
| ج) نمودار امتیاز اف-1 | |
| شکل 77- نمودارهای حاصل شده در فرایند آموزش شبکه SwinV2S الف) دقت ب) هزینه ج) امتیاز اف-1 | |

#### نتایج مدل ConvNextT در آزمون اول

شکل 4-15، نمودارهای حاصل شده از فرایند آموزش شبکه ConvNextT را نمایش می‌دهد. جدول 4-11، نتایج حاصل از ارزیابی این شبکه برروی داده‌های آزمون را نشان می‌دهد.

|  |  |
| --- | --- |
| جدول ‏4-11- درصد شاخص‌های عملکرد مدل ConvNextT برای هر برچسب | |
| |  |  |  |  | | --- | --- | --- | --- | | شاخص امتیاز اف-1 | شاخص صحت | شاخص یادآوری | نام برچسب | | 95.12 | 94.54 | 95.70 | **Scab** | | 99.62 | 99.35 | 99.89 | **Healthy** | | 91.62 | 91.83 | 91.41 | **Frog eye leaf spot** | | 97.69 | 98.77 | 96.63 | **Rust** | | 86.19 | 81.18 | 91.86 | **Complex** | | 97.59 | 99.18 | 96.04 | **Powdery mildew** | | |
|  | |
|  |  |
| الف) نمودار دقت | ب) نمودار هزینه |
|  | |
| ج) نمودار امتیاز اف-1 | |
| شکل 78- نمودارهای حاصل شده در فرایند آموزش شبکه ConvNextT الف) دقت ب) هزینه ج) امتیاز اف-1 | |

#### نتایج مدل ConvNextS در آزمون اول

شکل 4-16، نمودارهای حاصل شده از فرایند آموزش شبکه ConvNextS را نمایش می‌دهد. جدول 4-12، نتایج حاصل از ارزیابی این شبکه برروی داده‌های آزمون را نشان می‌دهد.

|  |  |
| --- | --- |
| جدول ‏4-12- درصد شاخص‌های عملکرد مدل ConvNextS برای هر برچسب | |
| |  |  |  |  | | --- | --- | --- | --- | | شاخص امتیاز اف-1 | شاخص صحت | شاخص یادآوری | نام برچسب | | 94.58 | 93.56 | 95.61 | **Scab** | | 99.41 | 98.83 | 100.00 | **Healthy** | | 90.27 | 88.20 | 92.44 | **Frog eye leaf spot** | | 96.70 | 98.26 | 95.19 | **Rust** | | 83.77 | 78.21 | 90.19 | **Complex** | | 98.60 | 99.59 | 97.62 | **Powdery mildew** | | |
|  | |
|  |  |
| الف) نمودار دقت | ب) نمودار هزینه |
|  | |
| ج) نمودار امتیاز اف-1 | |
| شکل 79- نمودارهای حاصل شده در فرایند آموزش شبکه ConvNextS الف) دقت ب) هزینه ج) امتیاز اف-1 | |

### آزمون دوم، بهینه‌سازی آستانه‌ی برچسب‌ها

در آزمون دوم با استفاده از الگوریتم‌های بهینه‌سازی بر پایه‌ی ازدحام، آستانه‌ی هر کدام از برچسب‌ها بهینه‌سازی می‌شوند تا دقت طبقه‌بندی شبکه‌ها افزایش یابد. در بخش3-7 توضیحاتی در مورد این نوع از الگوریتم‌های بهینه‌سازی داده شد. برای پیاده‌سازی این الگوریتم‌ها از کتاب‌خانه‌ Mealpy استفاده شده. از میان الگوریتم‌های بهینه‌سازی ازدحامی ارائه شده تسوط این کتاب‌خانه، 19 عدد از آن‌ها که دو متغیره هستند انتخاب و در کنار الگوریتم ژنتیک[[97]](#footnote-97) برای تغییر آستانه‌ها استفاده شده است. تمامی این الگوریتم‌ها نیاز به دو متغیر جمعیت و تعداد دوره دارند که به ترتیب 10 و 200 در نظر گرفته شده است. جدول 4-13 نتایج به دست آمده پس از تغییر دادن آستانه‌های هر برچسب توسط بهترین الگوریتم را نشان می‌دهد.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| جدول ‏4-13- نتایج به دست آمده پس از اعمال تغییرات یافته شده توسط الگوریتم‌های بهینه‌سازی ازدحامی | | | | |
| پس از بهینه‌سازی آستانه‌ها | | **قبل‌ از بهینه‌سازی آستانه‌ها** | |  |
| میانگین امتیاز اف-1 | **دقت** | **میانگین امتیاز اف-1** | **دقت** |  |
| 94.28 | 92.35 | 94.32 | 92.03 | **EfficientnetV2S** |
| 93.91 | 91.81 | 93.65 | 90.63 | **EfficientnetV2M** |
| 94.85 | 92.27 | 94.67 | 91.17 | **Resnet50** |
| 95.40 | 93.26 | 95.31 | 92.62 | **Resnext5032x4D** |
| 94.67 | 92.11 | 94.77 | 91.71 | **WideResnet50** |
| 93.92 | 91.87 | 93.83 | 91.41 | **RegnetY8GF** |
| 94.96 | 93.18 | 94.77 | 92.22 | **SwinV2T** |
| 94.88 | 93.00 | 94.75 | 92.51 | **SwinV2S** |
| 94.78 | 92.94 | 94.63 | 92.38 | **ConvNextT** |
| 94.12 | 92.32 | 93.88 | 91.17 | **ConvNextS** |

در فرایند بهینه‌سازی آستانه‌ها، تابع تناسب[[98]](#footnote-98) به گونه‌ای تعریف شده است که آستانه‌ی هر برچسب را از الگوریتم بهینه‌سازی دریافت کرده، و در خروجی نیز دقت مدل را ارائه می‌دهد. الگوریتم‌ها با تنظیم این 6 آستانه به دنبال بیشینه مطلق گشته و در نهایت بهترین آستانه‌های یافته شده را ارائه می‌دهند. با توجه به آزمون‌های اولیه انجام شده بر حدود آستانه‌ها، حد پایین مجاز برای تغییر این آستانه‌ها 0.1 و حد بالای آن 0.7 در نظر گرفته شده است. جدول 4-13، خلاصه مشخصات الگوریتم‌های مورد آزمون در تحقیق پیش رو را نمایش می‌دهد. جدول 4-14، الگوریتم‌های بهینه‌سازی که بهترین نتیجه را داشتند، برای هر مدل ارائه می‌دهد.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| جدول ‏4-14- مشخصات الگوریتم‌های بهینه‌سازی ازدحامی که در این تحقیق استفاده شده است | | | | |
| مرجع | **سال انتشار** | **نام فارسی الگوریتم** | **نام لاتین به اختصار** | **نام لاتین الگوریتم** |
| [116] | 2012 | الگوریتم بهینه‌سازی مگس میوه | FOA | **Fruit Fly Optimization Algorithm** |
| [117] | 2014 | بهینه‌ساز گرگ خاکستری | GWO | **Grey Wolf Optimizer** |
| [118] | 2015 | بهینه‌ساز مورچه‌گیر | ALO | **Ant Lion Optimizer** |
| [119] | 2015 | بهینه‌ساز شاپرک-شعله | MFO | **Moth Flame Optimization** |
| [120] | 2016 | الگوریتم جایا | JA | **Jaya Algorithm** |
| [121] | 2016 | الگوریتم بهینه‌سازی نهنگ | WOA | **Whale Optimization Algorithm** |
| [122] | 2016 | بهینه‌ساز سنجاقک | DO | **Dragonfly Optimization** |
| [123] | 2017 | بهینه‌سازی ازدحام سالپ | SSO | **Salp Swarm Optimization** |
| [124] | 2017 | ازدحام ربات‌های امداد | SRSR | **Swarm Robotics Search & Rescue** |
| [125] | 2019 | بهینه‌سازی شیر دریایی | SLO | **Sea Lion Optimization** |
| [126] | 2019 | الگوریتم مسیریاب | PFA | **Pathfinder Algorithm** |
| [127] | 2019 | بهینه‌سازی باز هَریس | HHO | **Harris Hawks Optimization** |
| [128] | 2021 | بهینه‌ساز عقاب | AO | **Aquila Optimizer** |
| [129] | 2022 | بهینه‌سازی ترکیبی گرگ-نهنگ | GWWO | **Hybrid Gray Wolf – Whale Optimization** |
| [130] | 2020 | الگوریتم شکارچیان دریایی | MPA | **Marine Predators Algorithm** |
| [131] | 2022 | الگوریتم گورکن عسل‌خوار | HBA | **Honey Badger Algorithm** |
| [132] | 2022 | بهینه‌سازی ازدحامی گربه‌شنی | SCSO | **Sand Cat Swarm Optimization** |
| [133] | 2021 | بهینه‌سازی ماهی تن | TSO | **Tuna Swarm Optimization** |
| [134] | 2022 | بهینه‌سازی خرگوش‌های مصنوعی | ARO | **Artificial Rabbits Optimization** |

|  |  |  |  |
| --- | --- | --- | --- |
| جدول ‏4-15- رتبه بندی بهترین الگوریتم‌های بهینه‌سازی بر اساس دقت مدل‌ها | | | |
| سومین الگوریتم بهینه‌سازی | **دومین الگوریتم بهینه‌سازی** | **بهترین الگوریتم بهینه‌سازی** | **نام مدل** |
| HBA | MPA | ARO | **EfficientnetV2S** |
| SSO | MPA | ARO | **EfficientnetV2M** |
| HBA | ARO | MPA | **Resnet50** |
| HBA | MPA | ARO | **Resnext5032x4D** |
| MPA | ARO | SSO | **WideResnet50** |
| TSO | SSO | MPA | **RegnetY8GF** |
| SLO | HBA | ARO | **SwinV2T** |
| SLO | ARO | HBA | **SwinV2S** |
| HBA | ARO | MPA | **ConvNextT** |
| SRSR | HBA | GA | **ConvNextS** |

با توجه به نتایج حاصل شده در جدول 4-15، الگوریتم‌هایی که در سال‌های اخیر طراحی و ارائه شدند، عملکرد بهتری نبست الگوریتم‌های قدیمی‌تر دارند. الگوریتم ژنتتیک نیز عملکرد قابل قبولی داشته‌است. در نهایت می‌توان گفت بهترین الگوریتم بهینه‌سازی برای حد آستانه مدل‌ها، ARO یا الگوریتم بهینه‌سازی خرگوش‌های مصنوعی می‌باشد. پس از آن و با اختلاف کمی، MPA یا الگوریتم بهینه‌سازی شکارچیان دریایی توانایی بهتری نسبت به سایر الگوریتم‌ها در تنظیم آستانه‌ها دارد. الگوریتم HBA یا گورکن عسل‌خوار نیز نتایج مطلوبی از خود نشان داده و مناسب این کار می‌باشد.

## نتیجه گیری

در ابتدای این فصل، مشخصات مجموعه‌داده مورد استفاده در این تحقیق ارائه و در مورد روش‌های رفع ایرادات آن بحث شد. سپس شرایط آموزش مدل‌های شبکه عصبی پیچشی ارائه شدند و در نهایت، نتایج حاصل از آموزش مدل‌ها شامل دقت، هزینه، امتیاز اف-1، صحت و یادآوری به دست آورده شد. در آزمونی دیگر، با استفاده از الگوریتم‌های بهینه‌سازی ازدحامی، حدود آستانه‌ی مناسب برای هر برچسب به صورت جداگانه یافت شد و افزایش دقت مدل‌ها را به همراه داشت. در نهایت، مقایسه‌ای بین این الگوریتم‌های بهینه‌سازی صورت گرفت و بهترین‌های آنها معرفی شد.

# فصل پنجم

نتیجه گیری و پیشنهادات

## مقدمه

در این تحقیق طبقه‌بندی تصاویر بیماری برگ گیاهان مجموعه‌داده مربوط به مجموعه‌داده Plant Pahology 2021، در 6 طبقه، به صورت چندبرچسبه و با استفاده از تکنیک یادگیری انتقالی و شبکه‌های عصبی پیچشی از پیش آموزش دیده‌شده مورد بررسی قرار گرفت. فصل اول به تشریح و تفسیر اهمیت و لزوم استفاده سامانه‌های هوشمند به جای روش‌های سنتی در جهت شناسائی بیماری‌های گیاهی پرداخت. در فصل دوم، مروری بر بیماری‌های رایج برگ درخت سیب و راه‌های مقابله با آنها ارائه شد. سپس تلاش‌های انجام‌شده جهت هوشمند‌سازی فرایند شناسایی بیماری‌ها با استفاده از هوش مصنوعی و پردازش تصویر یاد آوری شدند. در میان آنها، دو تحقیق بر روی مجموعه‌داده مورد بررسی انجام شده است. فصل سوم، وظیفه توضیح مواد و روش‌ها را به دوش کشیده است. در ابتدا روش‌های آموزش شبکه عصبی تفسیر شدند و سپس توضیحاتی در مورد اجزای شبکه‌های عصبی پیچشی بحث شد و برخی از معماری‌های آنها در کنار مبدلات تصاویر ارائه شد. فصل چهارم ابتدا به تشریح مجموعه‌داده مورد استفاده پرداخت و ایرادهای اساسی و روش‌های رفع آنها ارائه شدند. شرایط آموزش شبکه‌ها و تکنیک یادگیری انتقالی مشخص شد و در نهایت نتایج حاصل شده از آزمون اول بیان شدند. در آزمون دوم و با استفاده از الگوریتم‌های بهینه‌سازی ازدحامی آستانه شبکه‌ها تغییر داده شد، افزایش دقت مشاهده شده و در نهایت مقایسه‌ای بین این الگوریتم‌ها صورت گرفت. با گذر از این چهار فصل، فصل پنجم تحقیق حاضر با بیان مختصری از شرح عملکرد مدل‌های حاصل شده و برخی تغییراتی که تاثیری در آموزش آنها نداشته اند می‌پردازد و محدودیت‌ها و پیشنهادات تحقیقات بعدی را ارائه می‌کند.

## محتوا

در این بخش ابتدا به چند روش استفاده شده در این تحقیق که تاثیری بر افزایش دقت شبکه‌های آموزش دیده شده نداشتند اشاره می‌شود. سپس با جمع‌بندی تحقیقات انجام شده، نوآوری‌های صورت گرفته عنوان می‌شوند. بیان محدودیت‌های این تحقیق و پیشنهادهایی برای تحقیقات بعدی نیز بر عهده این بخش می‌باشد.

### تکنیک‌های استفاده شده دیگر

در این تحقیق از روش‌های رایج دیگری که در آموزش شبکه‌های عصبی پیچشی استفاده شده‌اند نیز استفاده شده است. اما این تکنیک‌ها باعث افزایش دقت طبقه‌بندی بیماری‌ها نشدند. در زیر به این تکنیک‌ها اشاره شده:

* تصاویر موجود در این مجموعه‌داده، به صورت Negative نیز به مدل‌های شبکه عصبی پیچشی داده شدند اما افزایش دقت مشاهده نشد.
* انواع دیگر فضا‌های رنگی مانند HSV و YCbCr برای تصاویر استفاده شدند اما هیچکدام باعث افزایش دقت مدل نشدند.
* استفاده از هر گونه تنظیم‌کننده نرخ یادگیری که باعث افزایش نرخ یادگیری در وسط و یا آخر فرآیند آموزش می‌شود، باعث افت شدید عملکرد و دقت مدل می‌شود. مانند تنظیم کننده نرخ یادگیری مثلثی[[99]](#footnote-99) .
* تصاویر موجود در مجموعه‌داده به دو روش تارسازی میانه‌ای[[100]](#footnote-100) و گوسی[[101]](#footnote-101) تار شدند و به شبکه‌ داده شدند اما افزایش دقتی حاصل نشد.
* از تکنیک برش-ترکیب در این تحقیق استفاده شد، با وجود آن که افزایش دقت در میان طبقه‌های شامل دو یا چند برچسب مشاهده شد، اما دقت کلی مدل در صورت استفاده از این تکنیک کاهش می‌یابد. نوع خاصی از این تکنیک استفاده شده است: از میان تصاویر400 در 400 موجود، برش‌هایی با ابعاد 180 در 180، 180 در 220، 220 در 180 و 220 در 220 از هر تصویر گرفته شده و در کنار هم قرار داده می‌شوند و سپس به شبکه داده می‌شوند. شکل 5-1 یک نمونه تصویر برش-ترکیب شده را نشان می‌دهد. برچسب‌های هر تصویر نیز با یکدیگر و به یک میزان مشارکت (100%) ترکیب می‌شوند. در حالتی که تصویری با برچسب بیماری و سالم تولید شود، برچسب بیماری آن حذف می‌شود.

|  |
| --- |
|  |
| شکل 80- نمونه‌ای از تصویر برش-ترکیب شده به همراه برچسب‌های آن |

### جمع‌بندی

با توجه به این که مجموعه‌داده مورد استفاده شامل 18633 تصویر از برگ‌ها می‌باشد و در مسائل طبقه‌بندی تصاویر، این عدد برای آموزش شبکه‌های عصبی پیچشی از ابتدا کم می‌باشد، به همین دلیل از تکنیک یادگیری انتقالی برای آموزش شبکه‌ها استفاده شد. در کنار آن، یادگیری انتقالی باعث همگراشدن سریعتر وزن‌های شبکه می‌شود و به این ترتیب، تعداد دوره‌های آموزشی کمتری لازم خواهد بود. 10 عدد از مدل‌های از پیش‌آموزش دیده شده بر روی مجموعه‌داده ImageNet از پایگاه پایتورچ دریافته شده و لایه آخر آنها به تعداد برچسب‌ها و اضافه شدن لایه برون‌انداز و تابع سیگموید تغییر پیدا می‌کنند. سپس فرآیند آموزش مدل‌ها آغاز می‌شود و به مدت 30 دوره و با کاهش نرخ یادگیری آموزش می‌بینند. بخشی از مجموعه‌داده به عنوان آزمون و جهت ارزیابی نهایی در نظر گرفته شده و دقت مدل‌ها و امتیاز اف-1 آنها مورد سنجش قرار می‌دهد. با توجه به نتایج آزمون‌، مدل‌های Resnext5032x4D و SwinV2S به ترتیب 92.62% و 92.51% بیشترین دقت و مدل‌های Resnext5032x4D و SwinV2T به ترتیب 95.31% و 94.77% بیشترین میانگین امتیاز اف-1 را داشتند. در آزمون دوم با استفاده از الگوریتم‌های بهینه‌سازی ازدحامی ابداع شده تا کنون، میزان آستانه‌ی برچسب‌های این مدل‌ها تغییر داده‌شد و دقت و امتیاز اف-1 آنها دوباره مورد سنجش قرار گرفتند. در این میان، مدل‌های Resnext5032x4D و SwinV2T پس از جایگزینی آستانه‌ها، به ترتیب بیشترین دقت 93.26% و 93.18% و بیشترین میانگین امتیاز اف-1 95.40% و 94.96% در طبقه‌بندی را داشتند. به دلیل آن که مجموعه‌داده مورد استفاده در این تحقیق، مجموعه‌داده جدیدی است، به نسبت تحقیق‌های کمتری بر روی آن صورت گرفته است. دو تحقیق [80] و [81] نیز از همین مجموعه‌داده استفاده کردند. مدل‌های شبکه‌ی عصبی پیچشی مورد بررسی در دو تحقیق یاد شده متفاوت و قدیمی‌تر بوده است.

## نوآوری

تحقیقات دیگری که در زمینه طبقه‌بندی بیماری‌های برگ گیاهان انجام شده است، عمدتا بر روی مجموعه‌داده PlantVillage بوده است و ایراد اصلی این مجموعه‌داده، نبودن شرایط محیطی و واقعی در تصاویر آموزشی است که در نهایت منجر به افت عملکرد تشخیص بیماری‌ها در شرایط واقعی می‌شود. تحقیقات متعدد دیگری نیز صورت گرفته است، اما در آنها از مجموعه‌داده‌های بسیار کوچک (بین 2000 تا 4000 تصویر) و یا مجموعه‌داده‌هایی که خصوصی هستند استفاده شده است. در میان آنها، مورد توجه‌ترین مورد [26] است که از مجموعه‌داده‌ی خصوصی PDD271 شامل 220 هزار تصویر از برگ گیاهان مختلف درختی و بوته‌ای در قالب 271 طبقه استفاده کرده است. دو تحقیقی که برروی مجموعه‌داده مورد استفاده در این تحقیق انجام شده است، از مدل‌های دیگری استفاده کرده‌اند و این مجموعه‌داده را به صورت مسئله طبقه‌بندی چندطبقه در نظر گرفته اند. در تحقیق حاضر از مدل‌های شبکه عصبی پیچشی از پیش آموزش دیده شده جدیدتر که دو تحقیق یاد شده مورد بررسی قرار نداده‌اند، استفاده شده است. همچنین تغییراتی در ساختار مدل‌ها و روند آموزش آنها (مانند بلوک برون‌انداز) در نظر گرفته شده است که باعث افزایش عملکرد مدل‌ها می‌شود. در تحقیق حاضر، مسئله طبقه‌بندی به صورت چندبرچسبه در نظر گرفته شده است و امکان شناسائی چند بیماری که ترکیب آنها در میان طبقه‌ها نیستند نیز وجود دارد. در کنار تمام ویژگی‌های این تحقیق، مقایسه‌ای بین الگوریتم‌های بهینه‌سازی ازدحامی در راستای تنظیم حد آستانه برچسب‌ها صورت گرفته است که همه آنها افزایش دقت را به همراه داشته اند.

### محدودیت‌ها

معماری‌های یادگیری عمیق برای یادگیری الگو‌های مفید به تعداد زیادی داده‌ی آموزشی نیاز دارند. ایجاد مجموعه‌داده‌ای وسیع از تصاویر بیماری برگ گیاهان نیازمند زمان و هزینه بالایی می‌باشد، چرا که نیازمند یافتن محیط‌های مختلف جغرافیایی و شرایط نوری متنوعی هستند و همه این تصاویر باید توسط متخصصان برچسب‌گذاری شوند. همچنین یافتن تصاویری که به صورت طبیعی چند بیماری روی برگ دارند، امری بسیار دشوار می‌باشد. مدل‌های آموزش‌دیده شده تنها از مجموعه‌داده‌ی Plant Pathology 2021 جهت آموزش، اعتبارسنجی و آزمون استفاده کرده‌اند. نبود تصاویر زمین‌های کشاورزی در مناطق و شرایط آب و هوایی مختلف در سرتاسر جهان و کمیاب بودن تصاویر برگ شامل چند بیماری باعث محدود شدن آموزش شبکه‌ها می‌شود.

محدودیت دیگر، منابع سخت‌افزاری استفاده شده در این تحقیق می‌باشد. سامانه‌های تحت وب مانند گوگل کولب[[102]](#footnote-102) و کگل نوت بوک[[103]](#footnote-103) وجود دارد که به محققین مایل به آموزش شبکه‌عصبی، سخت‌افزار به صورت رایگان ارائه می‌دهند. اما این سامانه‌ها محدودیت استفاده در روز و هفته دارند یا امکان قطع شدن اتصال آنها وجود دارد و مناسب فرآیندهای آموزشی طولانی نیستند. رایانه شخصی استفاده شده در این تحقیق، علاوه بر نداشتن این محدودیت‌ها از سخت‌افزارهای ارائه شده توسط این سامانه‌ها قدرتمندتر بوده و فرآیند آموزش شبکه در آن سریع‌تر انجام می‌شود. اگرچه گمان می‌رود استفاده از مدل‌های حجیم‌تر و سنگین‌تر ممکن است باعث پدیده بیش‌برآزش شود، اما برای اثبات این موضوع انجام آزمایش‌ها بر روی سخت افزار‌های بهتر لازم است.

دیگر محدودیت تحقیق حاضر این است که مجموعه‌داده استفاده شده، با دقت بالا برچسب‌گذاری نشده است. تصاویری وجود دارد که به نظر می‌رسد برچسب متفاوتی مناسب آنها باشند یا تصاویری هستند که دوبار از آنها عکس‌برداری شده است. همچنین تصاویری با وضعیت نامشخص و یا حضور دو برگ متفاوت وجود دارند که باعث کاهش توانایی طبقه‌بندی شبکه‌ها می‌شوند. در شکل 5-2 نمونه‌ای از این تصاویر نمایش داده شده. برخی از تصاویر این مجموعه‌داده نیازمند تجدید نظر بوده و بهتر است این پایگاه‌داده توسط متخصصین بازبینی شود.

|  |  |  |
| --- | --- | --- |
|  |  |  |
| شکل 81- نمونه‌ای از تصاویر نامشخص موجود در مجموعه‌داده | | |

### پیشنهادها

در امتداد این تحقیق، پیشنهاد‌هایی برای کار‌های آینده توصیه می‌شود. اول، در راستای گردآوری مجموعه‌داده جامع برای گیاهان دیگر و بیماری‌های جدید اقداماتی صورت گیرد تا بتوان این مدل‌های شبکه عصبی پیچشی را بیشتر آموزش داده و اعتبارسنجی نمود. برای آموزش بهتر این مدل‌ها می‌توان از تصاویری از برگ‌های مزارعی که در مناطق جغرافیایی مختلف قرار دارند تهیه نمود. روش دیگری که می‌توان برای افزایش تعداد تصاویر ورودی بهره برد، استفاده از شبکه‌های عصبی مولد تخاصمی[[104]](#footnote-104) است. می‌توان تصاویری برگ‌های سالم را به این شبکه‌ها داده و تصویر برگ شامل بیماری تولید کرد. سپس با ترکیب تصاویر تولیدشده، امکان ساخت تصاویر شامل چند بیماری به وجود می‌آید که می‌تواند عدم توازن داده‌ها را بسیار بهبود ببخشد. البته باید توجه داشت که این شبکه‌ها منابع سخت‌افزاری بسیار قوی تر از آموزش شبکه‌های پیچشی نیاز دارند.

دوم می‌توان با ایده‌های نوآورانه و روش‌های نوین، تلاش به جهت ارتقای عملکرد مدل‌ها صورت گیرد. می‌توان با ترکیب مدل‌ها با یک دیگر و ایجاد شبکه‌های عصبی ترکیبی، دقت سامانه را به میزان کمی بالا برد. در کنار آن، تکنیک‌های آموزش و دیگر و فضا‌های رنگی دیگر گزینه مناسبی برای کنکاش و آزمایش هستند که ممکن است عملکرد سامانه را بهبود بخشد.

سوم پس از آزمودن کامل سامانه و عملکرد آن، بازاریابی و مذاکرات لازم با وزارت جهاد کشاورزی و انجمن‌های باغبانی ایران صورت گیرد تا مدل نهایی را به بخش دولتی واگذار کرد. همچنین امکان پیاده‌سازی مدل بر روی دستگاه‌های جتسون نانو[[105]](#footnote-105) و سوار کردن آن بر روی پهپادها و یا ربات‌های کوچک وجود دارد و با تنظیم آن‌ها می‌توان به صورت خودکار کل زمین کشاورزی مورد بررسی قرار گیرد. از میان مدل‌های آزموده‌شده، تنها مدل Resnext5032x4D عملکرد مناسب و حجم کمی دارد که مناسب پیاده‌سازی بر روی دستگاه‌های تلفن همراه می‌باشد.

چهارم می‌توان با استفاده از روش چندبرچسبه و آستانه‌گذاری برچسب‌ها سامانه‌ای تدارک دید که توانایی شناسایی برگ‌های مشکوک به یک بیماری را دارند. به این صورت که دو حد آستانه برای مدل شبکه عصبی تعریف شود، یکی حد پایین که اگر خروجی برچسبی کمتر از آن بود، عدم وجود آن را گزارش دهد. حد آستانه‌ی بالا نیز به گونه‌ای است که اگر خروجی آن برچسب بیشتر باشد، تصویر بررسی شده آن بیماری را دارد. حال اگر خروجی یک برچسب بین این دو حد آستانه قرار بگیرد، می‌توان گفت که این برگ مشکوک به بیماری مورد بررسی است. تصاویری که این ویژگی را داشته باشند، می‌توان به متخصصین گیاه‌پزشکی نشان داده تا وضعیت نهایی بیماری مشخص شود. با این روش امکان طراحی سامانه‌ای که توانایی یادگیری از تصاویر جدید دارد فراهم می‌شود.

# مراجع

[1] M. Loey, A. ElSawy, and M. Afify, “Deep learning in plant diseases detection for agricultural crops: A survey,” *Int. J. Serv. Sci. Manag. Eng. Technol.*, vol. 11, no. 2, pp. 41–58, Apr. 2020, doi: 10.4018/IJSSMET.2020040103.

[2] A. Darwish, D. Ezzat, and A. E. Hassanien, “An optimized model based on convolutional neural networks and orthogonal learning particle swarm optimization algorithm for plant diseases diagnosis,” *Swarm Evol. Comput.*, vol. 52, p. 100616, Feb. 2020, doi: 10.1016/j.swevo.2019.100616.

[3] U. N. D. of E. and S. A. P. Division, “World population prospects 2019,” *Dep. Econ. Soc. Aff. World Popul. Prospect. 2019.*, no. 141, 2019.

[4] D. Radovanovic and S. Dukanovic, “Image-Based Plant Disease Detection: A Comparison of Deep Learning and Classical Machine Learning Algorithms,” in *2020 24th International Conference on Information Technology, IT 2020*, IEEE, Feb. 2020, pp. 1–4. doi: 10.1109/IT48810.2020.9070664.

[5] I. Z. Mukti and D. Biswas, “Transfer Learning Based Plant Diseases Detection Using ResNet50,” in *2019 4th International Conference on Electrical Information and Communication Technology, EICT 2019*, IEEE, Dec. 2019, pp. 1–6. doi: 10.1109/EICT48899.2019.9068805.

[6] J. G. A. Barbedo, “A review on the main challenges in automatic plant disease identification based on visible range images,” *Biosyst. Eng.*, vol. 144, pp. 52–60, Apr. 2016, doi: 10.1016/j.biosystemseng.2016.01.017.

[7] S. P. Mohanty, D. P. Hughes, and M. Salathé, “Using deep learning for image-based plant disease detection,” *Front. Plant Sci.*, vol. 7, no. September, Sep. 2016, doi: 10.3389/fpls.2016.01419.

[8] J. G. A. Barbedo, “Factors influencing the use of deep learning for plant disease recognition,” *Biosyst. Eng.*, vol. 172, pp. 84–91, Aug. 2018, doi: 10.1016/j.biosystemseng.2018.05.013.

[9] C. H. Bock, G. H. Poole, P. E. Parker, and T. R. Gottwald, “Plant disease severity estimated visually, by digital photography and image analysis, and by hyperspectral imaging,” *CRC. Crit. Rev. Plant Sci.*, vol. 29, no. 2, pp. 59–107, Mar. 2010, doi: 10.1080/07352681003617285.

[10] J. G. Arnal Barbedo, “Digital image processing techniques for detecting, quantifying and classifying plant diseases,” *Springerplus*, vol. 2, no. 1, pp. 1–12, Dec. 2013, doi: 10.1186/2193-1801-2-660.

[11] G. L. Grinblat, L. C. Uzal, M. G. Larese, and P. M. Granitto, “Deep learning for plant identification using vein morphological patterns,” *Comput. Electron. Agric.*, vol. 127, pp. 418–424, Sep. 2016, doi: 10.1016/j.compag.2016.07.003.

[12] A. Ramcharan, K. Baranowski, P. McCloskey, B. Ahmed, J. Legg, and D. P. Hughes, “Deep learning for image-based cassava disease detection,” *Front. Plant Sci.*, vol. 8, Oct. 2017, doi: 10.3389/fpls.2017.01852.

[13] L. Li, S. Zhang, and B. Wang, “Plant Disease Detection and Classification by Deep Learning - A Review,” *IEEE Access*, vol. 9, pp. 56683–56698, 2021, doi: 10.1109/ACCESS.2021.3069646.

[14] A. Johannes *et al.*, “Automatic plant disease diagnosis using mobile capture devices, applied on a wheat use case,” *Comput. Electron. Agric.*, vol. 138, pp. 200–209, Jun. 2017, doi: 10.1016/j.compag.2017.04.013.

[15] M. C. Enebe and O. O. Babalola, “The impact of microbes in the orchestration of plants’ resistance to biotic stress: a disease management approach,” *Appl. Microbiol. Biotechnol.*, vol. 103, no. 1, pp. 9–25, Jan. 2019, doi: 10.1007/s00253-018-9433-3.

[16] S. Savary, L. Willocquet, S. J. Pethybridge, P. Esker, N. McRoberts, and A. Nelson, “The global burden of pathogens and pests on major food crops,” *Nat. Ecol. Evol.*, vol. 3, no. 3, pp. 430–439, Feb. 2019, doi: 10.1038/s41559-018-0793-y.

[17] A. J. Tatem, D. J. Rogers, and S. I. Hay, “Global Transport Networks and Infectious Disease Spread,” in *Advances in Parasitology*, 2006, pp. 293–343. doi: 10.1016/S0065-308X(05)62009-X.

[18] J. R. Rohr, T. R. Raffel, J. M. Romansic, H. McCallum, and P. J. Hudson, “Evaluating the links between climate, disease spread, and amphibian declines,” *Proc. Natl. Acad. Sci. U. S. A.*, vol. 105, no. 45, pp. 17436–17441, Nov. 2008, doi: 10.1073/pnas.0806368105.

[19] T. van der Zwet, “Present Worldwide Distribution of Fire Blight,” *Acta Hortic.*, no. 590, pp. 33–34, Nov. 2002, doi: 10.17660/actahortic.2002.590.1.

[20] S. A. Miller, F. D. Beed, and C. L. Harmon, “Plant disease diagnostic capabilities and networks,” *Annu. Rev. Phytopathol.*, vol. 47, no. 1, pp. 15–38, Sep. 2009, doi: 10.1146/annurev-phyto-080508-081743.

[21] H. Al Hiary, S. Bani Ahmad, M. Reyalat, M. Braik, and Z. ALRahamneh, “Fast and Accurate Detection and Classification of Plant Diseases,” *Int. J. Comput. Appl.*, vol. 17, no. 1, pp. 31–38, Mar. 2011, doi: 10.5120/2183-2754.

[22] X. Bai *et al.*, “Rice heading stage automatic observation by multi-classifier cascade based rice spike detection method,” *Agric. For. Meteorol.*, vol. 259, pp. 260–270, Sep. 2018, doi: 10.1016/j.agrformet.2018.05.001.

[23] K. P. Ferentinos, “Deep learning models for plant disease detection and diagnosis,” *Comput. Electron. Agric.*, vol. 145, pp. 311–318, Feb. 2018, doi: 10.1016/j.compag.2018.01.009.

[24] R. Thapa, K. Zhang, N. Snavely, S. Belongie, and A. Khan, “The Plant Pathology Challenge 2020 data set to classify foliar disease of apples,” *Appl. Plant Sci.*, vol. 8, no. 9, Sep. 2020, doi: 10.1002/aps3.11390.

[25] J. G. Arnal Barbedo, “Plant disease identification from individual lesions and spots using deep learning,” *Biosyst. Eng.*, vol. 180, pp. 96–107, Apr. 2019, doi: 10.1016/j.biosystemseng.2019.02.002.

[26] X. Liu, W. Min, S. Mei, L. Wang, and S. Jiang, “Plant Disease Recognition: A Large-Scale Benchmark Dataset and a Visual Region and Loss Reweighting Approach,” *IEEE Trans. Image Process.*, vol. 30, pp. 2003–2015, 2021, doi: 10.1109/TIP.2021.3049334.

[27] R. Thapa, K. Zhang, N. Snavely, S. Belongie, and A. Khan, “The Plant Pathology Challenge 2020 data set to classify foliar disease of apples,” *Appl. Plant Sci.*, vol. 8, no. 9, Apr. 2020, doi: 10.1002/aps3.11390.

[28] W. Albattah, M. Nawaz, A. Javed, M. Masood, and S. Albahli, “A novel deep learning method for detection and classification of plant diseases,” *Complex Intell. Syst.*, vol. 8, no. 1, pp. 507–524, Feb. 2022, doi: 10.1007/s40747-021-00536-1.

[29] L. Liu *et al.*, “Deep Learning for Generic Object Detection: A Survey,” *Int. J. Comput. Vis.*, vol. 128, no. 2, pp. 261–318, Feb. 2020, doi: 10.1007/s11263-019-01247-4.

[30] M. Goyal, N. D. Reeves, A. K. Davison, S. Rajbhandari, J. Spragg, and M. H. Yap, “DFUNet: Convolutional Neural Networks for Diabetic Foot Ulcer Classification,” *IEEE Trans. Emerg. Top. Comput. Intell.*, vol. 4, no. 5, pp. 728–739, Oct. 2018, doi: 10.1109/tetci.2018.2866254.

[31] A. Khan, A. Sohail, U. Zahoora, and A. S. Qureshi, “A survey of the recent architectures of deep convolutional neural networks,” *Artif. Intell. Rev.*, vol. 53, no. 8, pp. 5455–5516, Dec. 2020, doi: 10.1007/s10462-020-09825-6.

[32] Tee KP, Ge SS, Tay EH (2009) Barrier Lyapunov Functions for

the control of output-constrained nonlinear systems. Automatica

45(4):918–927. https://doi.org/10.1016/j.automatica.2008.11.017

J. Carranza-Rojas, H. Goeau, P. Bonnet, E. Mata-Montero, and A. Joly, “Going deeper in the automated identification of Herbarium specimens,” *BMC Evol. Biol.*, vol. 17, no. 1, p. 181, Dec. 2017, doi: 10.1186/s12862-017-1014-z.

[33] X. Yang and T. Guo, “Machine learning in plant disease research,” *Eur. J. Biomed. Res.*, vol. 3, no. 1, p. 6, Mar. 2017, doi: 10.18088/ejbmr.3.1.2017.pp6-9.

[34] J. G. A. Barbedo, “Impact of dataset size and variety on the effectiveness of deep learning and transfer learning for plant disease classification,” *Comput. Electron. Agric.*, vol. 153, pp. 46–53, Oct. 2018, doi: 10.1016/j.compag.2018.08.013.

[35] M. C. Press, “The functional significance of leaf structure: A search for generalizations,” *New Phytol.*, vol. 143, no. 1, pp. 213–219, Jul. 1999, doi: 10.1046/j.1469-8137.1999.00432.x.

[36] S. Baas, S. Conforti, S. Ahmed, and G. Markova, “The impact of disasters and crises on agriculture and food security: 2021,” *impact disasters Cris. Agric. food Secur. 2021*, 2021, doi: 10.4060/cb3673en.

[37] Y. Toda and F. Okura, “How convolutional neural networks diagnose plant disease,” *Plant Phenomics*, vol. 2019, Jan. 2019, doi: 10.34133/2019/9237136.

[38] J. Boulent, S. Foucher, J. Théau, and P. L. St-Charles, “Convolutional Neural Networks for the Automatic Identification of Plant Diseases,” *Front. Plant Sci.*, vol. 10, Jul. 2019, doi: 10.3389/fpls.2019.00941.

[39] W. E. MacHardy, “Apple Scab: Biology, Epidemiology, and Management,” *St. Paul, MN Am. Phytopathol. Soc. Press.*, 1996.

[40] D. W. L. Manktelow, R. M. Beresford, T. A. Batchelor, and J. T. S. Walker, “Use patterns and economics of fungicides for disease control in New Zealand apples,” *Acta Hortic.*, vol. 422, no. 422, pp. 187–192, Jul. 1996, doi: 10.17660/ActaHortic.1996.422.31.

[41] O. Carisse and J. Bernier, “Effect of environmental factors on growth, pycnidial production and spore germination of Microsphaeropsis isolates with biocontrol potential against apple scab,” *Mycol. Res.*, vol. 106, no. 12, pp. 1455–1462, Dec. 2002, doi: 10.1017/S0953756202006858.

[42] R. M. Beresford and D. W. L. Manktelow, “Economics of reducing fungicide use by weather-based disease forecasts for control of venturis inaequalis in apples,” *New Zeal. J. Crop Hortic. Sci.*, vol. 22, no. 2, pp. 113–120, Jun. 1994, doi: 10.1080/01140671.1994.9513814.

[43] A. L. Jones and T. B. Sutton, “Diseases of Tree Fruits in the East,” *Coop. Ext. Serv. Michigan State Univ. East Lansing.*, p. 98, 1996.

[44] C. E. Main and Gurtz, S., “1987 Crop lossses in North Carolina due to plant diseases and nematodes,” *N. C. State Univ. Dep. Plant Pathol. Spec. Publ.*, vol. 7, p. 183, 1988.

[45] P. M. Miller, “Piles of Apple Prunings as Sources of Conidia of Physalospora obtusa,” *Phytopathology*, vol. 63, no. 8, p. 1080, 1973, doi: 10.1094/phyto-63-1080.

[46] D. Strickland, J. Carrol, and K. Cox, “Cedar Apple Rust,” *New York State Integr. Pest Manag. Program, Subj. Agric. IPM; Fruits; Tree Fruit; Apples*, 2020.

[47] M. A. Ellis, “ Photosynthesis, Transpiration, and Carbohydrate Content of Apple Leaves Infected by Podosphaera leucotricha ,” *Phytopathology*, vol. 71, no. 4, p. 392, 1981, doi: 10.1094/phyto-71-392.

[48] X. M. Xu, “Modelling and forecasting epidemics of apple powdery mildew (Podosphaera leucotricha),” *Plant Pathol.*, vol. 48, no. 4, pp. 462–471, Aug. 1999, doi: 10.1046/j.1365-3059.1999.00371.x.

[49] T. B. Sutton, H. S. Aldwinckle, A. M. Angello, and J. F. Walgenbach, *Compendium of Apple and Pear Diseases and Pests, Second Edition*. The American Phytopathological Society, 2014. doi: 10.1094/9780890544334.

[50] A. Urbanietz and F. Dunemann, “Isolation, identification and molecular characterization of physiological races of apple powdery mildew (Podosphaera leucotricha),” *Plant Pathol.*, vol. 54, no. 2, pp. 125–133, Apr. 2005, doi: 10.1111/j.1365-3059.2005.01156.x.

[51] S. R. Maniyath *et al.*, “Plant disease detection using machine learning,” in *Proceedings - 2018 International Conference on Design Innovations for 3Cs Compute Communicate Control, ICDI3C 2018*, IEEE, Apr. 2018, pp. 41–45. doi: 10.1109/ICDI3C.2018.00017.

[52] W. K. Pratt, “Digital Image Processing,” *Eur. J. Eng. Educ.*, vol. 19, no. 3, p. 377, 1994, doi: 10.1080/03043799408928319.

[53] R. Gonzalez and R. Woods, “A text book on ‘Digital Image Processing,’” *Publ. Pearson*, vol. 2nd Editio, 2002.

[54] S. Phadikar and J. Sil, “Rice disease identification using pattern recognition techniques,” in *Proceedings of 11th International Conference on Computer and Information Technology, ICCIT 2008*, IEEE, Dec. 2008, pp. 420–423. doi: 10.1109/ICCITECHN.2008.4803079.

[55] D. Al Bashish, M. Braik, and S. Bani-Ahmad, “A framework for detection and classification of plant leaf and stem diseases,” in *Proceedings of the 2010 International Conference on Signal and Image Processing, ICSIP 2010*, IEEE, Dec. 2010, pp. 113–118. doi: 10.1109/ICSIP.2010.5697452.

[56] S. H. Lee, H. Goëau, P. Bonnet, and A. Joly, “New perspectives on plant disease characterization based on deep learning,” *Comput. Electron. Agric.*, vol. 170, p. 105220, Mar. 2020, doi: 10.1016/j.compag.2020.105220.

[57] A. Krizhevsky, I. Sutskever, and G. E. Hinton, “ImageNet classification with deep convolutional neural networks,” *Commun. ACM*, vol. 60, no. 6, pp. 84–90, May 2017, doi: 10.1145/3065386.

[58] M. Loey, F. Smarandache, and N. E. M. Khalifa, “Within the lack of chest COVID-19 X-ray dataset: A novel detection model based on GAN and deep transfer learning,” *Symmetry (Basel).*, vol. 12, no. 4, p. 651, Apr. 2020, doi: 10.3390/SYM12040651.

[59] D. Rong, L. Xie, and Y. Ying, “Computer vision detection of foreign objects in walnuts using deep learning,” *Comput. Electron. Agric.*, vol. 162, pp. 1001–1010, Jul. 2019, doi: 10.1016/j.compag.2019.05.019.

[60] G. Eraslan, Ž. Avsec, J. Gagneur, and F. J. Theis, “Deep learning: new computational modelling techniques for genomics,” *Nat. Rev. Genet.*, vol. 20, no. 7, pp. 389–403, Jul. 2019, doi: 10.1038/s41576-019-0122-6.

[61] Y. Lecun, Y. Bengio, and G. Hinton, “Deep learning,” *Nature*, vol. 521, no. 7553, pp. 436–444, May 2015, doi: 10.1038/nature14539.

[62] Y. Peng *et al.*, “DeepSeeNet: A Deep Learning Model for Automated Classification of Patient-based Age-related Macular Degeneration Severity from Color Fundus Photographs,” *Ophthalmology*, vol. 126, no. 4, pp. 565–575, Apr. 2019, doi: 10.1016/j.ophtha.2018.11.015.

[63] I. G. and Y. B. and A. Courville, “Deep learning 简介 一 、 什么是 Deep Learning ？,” *Nature*, vol. 29, no. 7553, pp. 1–73, 2016, [Online]. Available: http://deeplearning.net/

[64] J. P. Shah, H. B. Prajapati, and V. K. Dabhi, “A survey on detection and classification of rice plant diseases,” in *2016 IEEE International Conference on Current Trends in Advanced Computing, ICCTAC 2016*, IEEE, Mar. 2016, pp. 1–8. doi: 10.1109/ICCTAC.2016.7567333.

[65] H. B. Prajapati, J. P. Shah, and V. K. Dabhi, “Detection and classification of rice plant diseases,” *Intell. Decis. Technol.*, vol. 11, no. 3, pp. 357–373, Aug. 2017, doi: 10.3233/IDT-170301.

[66] J. G. A. Barbedo *et al.*, “Annotated plant pathology databases for image-based detection and recognition of diseases,” *IEEE Lat. Am. Trans.*, vol. 16, no. 6, pp. 1749–1757, Jun. 2018, doi: 10.1109/TLA.2018.8444395.

[67] M. Brahimi, M. Arsenovic, S. Laraba, S. Sladojevic, K. Boukhalfa, and A. Moussaoui, “Deep Learning for Plant Diseases: Detection and Saliency Map Visualisation,” 2018, pp. 93–117. doi: 10.1007/978-3-319-90403-0\_6.

[68] T. Wiesner-Hanks *et al.*, “Image set for deep learning: Field images of maize annotated with disease symptoms,” *BMC Res. Notes*, vol. 11, no. 1, p. 440, Dec. 2018, doi: 10.1186/s13104-018-3548-6.

[69] yuanyuan@iim.ac.cn and chenlei@iim.ac.cn, “Image Caption Machine Learning Dataset,” *http://www.icgroupcas.cn/website\_bchtk/index.html*, 2018.

[70] D. Singh, N. Jain, P. Jain, P. Kayal, S. Kumawat, and N. Batra, “PlantDoc: A dataset for visual plant disease detection,” in *ACM International Conference Proceeding Series*, New York, NY, USA: ACM, Jan. 2020, pp. 249–253. doi: 10.1145/3371158.3371196.

[71] Makerere University AI Lab, “Cassava Leaf Disease Classification,” *Kaggle* , 2021, [Online]. Available: https://www.kaggle.com/c/cassava-leaf-disease-classification

[72] J. Laaksonen, M. Koskela, and E. Oja, “PicSOM: Self-Organizing Maps for content-based image retrieval,” in *Proceedings of the International Joint Conference on Neural Networks*, IEEE, 1999, pp. 2470–2473. doi: 10.1109/ijcnn.1999.833459.

[73] G. Wang, Y. Sun, and J. Wang, “Automatic Image-Based Plant Disease Severity Estimation Using Deep Learning,” *Comput. Intell. Neurosci.*, vol. 2017, pp. 1–8, 2017, doi: 10.1155/2017/2917536.

[74] J. Graham, K. Clare, C. Bill, B. Rosie, and T. Lindy, “the Encyclopaedia of Cereal Diseases,” *Encycl. Cereal Dis.*, pp. 42–72, 2008, [Online]. Available: http://www.agricentre.basf.co.uk/agroportal/uk/media/marketing\_pages/cereal\_fungicides/BASF\_Disease\_Encyclopedia.pdf

[75] E. C. Too, L. Yujian, S. Njuki, and L. Yingchun, “A comparative study of fine-tuning deep learning models for plant disease identification,” *Comput. Electron. Agric.*, vol. 161, pp. 272–279, Jun. 2019, doi: 10.1016/j.compag.2018.03.032.

[76] S. K. Noon, M. Amjad, M. A. Qureshi, and A. Mannan, “Overfitting Mitigation Analysis in Deep Learning Models for Plant Leaf Disease Recognition,” in *Proceedings - 2020 23rd IEEE International Multi-Topic Conference, INMIC 2020*, IEEE, Nov. 2020, pp. 1–5. doi: 10.1109/INMIC50486.2020.9318044.

[77] R. Hummel, “Image Enhancement By Histogram Transformation.,” *Comput Graph. Image Process*, vol. 6, no. 2, pp. 184–185, Apr. 1977, doi: 10.1016/s0146-664x(77)80011-7.

[78] G. Yadav, S. Maheshwari, and A. Agarwal, “Contrast limited adaptive histogram equalization based enhancement for real time video system,” in *Proceedings of the 2014 International Conference on Advances in Computing, Communications and Informatics, ICACCI 2014*, IEEE, Sep. 2014, pp. 2392–2397. doi: 10.1109/ICACCI.2014.6968381.

[79] G. Sambasivam and G. D. Opiyo, “A predictive machine learning application in agriculture: Cassava disease detection and classification with imbalanced dataset using convolutional neural networks,” *Egypt. Informatics J.*, vol. 22, no. 1, pp. 27–34, Mar. 2021, doi: 10.1016/j.eij.2020.02.007.

[80] K. Vora and D. Padalia, “An Ensemble of Convolutional Neural Networks to Detect Foliar Diseases in Apple Plants,” Oct. 2021.

[81] A. Yadav, U. Thakur, R. Saxena, V. Pal, V. Bhateja, and J. C. W. Lin, “AFD-Net: Apple Foliar Disease multi classification using deep learning on plant pathology dataset,” *Plant Soil*, vol. 477, no. 1–2, pp. 595–611, Aug. 2022, doi: 10.1007/s11104-022-05407-3.

[82] S. S. Chouhan, U. P. Singh, A. Kaul, and S. Jain, “A Data Repository of Leaf Images: Practice towards Plant Conservation with Plant Pathology,” in *2019 4th International Conference on Information Systems and Computer Networks, ISCON 2019*, IEEE, Nov. 2019, pp. 700–707. doi: 10.1109/ISCON47742.2019.9036158.

[83] S. Parez, N. Dilshad, T. M. Alanazi, and J. Weon Lee, “Towards Sustainable Agricultural Systems: A Lightweight Deep Learning Model for Plant Disease Detection,” *Comput. Syst. Sci. Eng.*, vol. 47, no. 1, pp. 515–536, 2023, doi: 10.32604/csse.2023.037992.

[84] T. A. Prasetyo, V. L. Desrony, H. F. Panjaitan, R. Sianipar, and Y. Pratama, “Corn plant disease classification based on leaf using residual networks-9 architecture,” *Int. J. Electr. Comput. Eng.*, vol. 13, no. 3, pp. 2908–2920, Jun. 2023, doi: 10.11591/ijece.v13i3.pp2908-2920.

[85] S. U. P. U. Stanford Vision Lab, “http://www.image-net.org/,” 2014.

[86] C. Szegedy, V. Vanhoucke, S. Ioffe, J. Shlens, and Z. Wojna, “Rethinking the Inception Architecture for Computer Vision,” in *Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, IEEE, Jun. 2016, pp. 2818–2826. doi: 10.1109/CVPR.2016.308.

[87] K. Simonyan and A. Zisserman, “Very deep convolutional networks for large-scale image recognition,” *3rd Int. Conf. Learn. Represent. ICLR 2015 - Conf. Track Proc.*, Sep. 2015.

[88] M. Tan and Q. V. Le, “EfficientNet: Rethinking model scaling for convolutional neural networks,” *36th Int. Conf. Mach. Learn. ICML 2019*, vol. 2019-June, pp. 10691–10700, May 2019.

[89] K. He, X. Zhang, S. Ren, and J. Sun, “Deep residual learning for image recognition,” in *Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, IEEE, Jun. 2016, pp. 770–778. doi: 10.1109/CVPR.2016.90.

[90] Z. Liu *et al.*, “Swin Transformer: Hierarchical Vision Transformer using Shifted Windows,” in *Proceedings of the IEEE International Conference on Computer Vision*, IEEE, Oct. 2021, pp. 9992–10002. doi: 10.1109/ICCV48922.2021.00986.

[91] S. Cuenat and R. Couturier, “Convolutional Neural Network (CNN) vs Vision Transformer (ViT) for Digital Holography,” *2022 2nd Int. Conf. Comput. Control Robot. ICCCR 2022*, pp. 235–240, Aug. 2022, doi: 10.1109/ICCCR54399.2022.9790134.

[92] I. Radosavovic, R. P. Kosaraju, R. Girshick, K. He, and P. Dollár, “Designing network design spaces,” in *Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, IEEE, Jun. 2020, pp. 10425–10433. doi: 10.1109/CVPR42600.2020.01044.

[93] M. I. Jordan and T. M. Mitchell, “Machine learning: Trends, perspectives, and prospects,” *Science (80-. ).*, vol. 349, no. 6245, pp. 255–260, Jul. 2015, doi: 10.1126/science.aaa8415.

[94] R. M. Neal, *Pattern Recognition and Machine Learning*, vol. 49, no. 3. Springer, 2007. doi: 10.1198/tech.2007.s518.

[95] J. Hu, H. Niu, J. Carrasco, B. Lennox, and F. Arvin, “Voronoi-Based Multi-Robot Autonomous Exploration in Unknown Environments via Deep Reinforcement Learning,” *IEEE Trans. Veh. Technol.*, vol. 69, no. 12, pp. 14413–14423, Dec. 2020, doi: 10.1109/TVT.2020.3034800.

[96] M. Yoosefzadeh-Najafabadi, H. J. Earl, D. Tulpan, J. Sulik, and M. Eskandari, “Application of Machine Learning Algorithms in Plant Breeding: Predicting Yield From Hyperspectral Reflectance in Soybean,” *Front. Plant Sci.*, vol. 11, Jan. 2021, doi: 10.3389/fpls.2020.624273.

[97] R. Féraud and F. Clérot, “A methodology to explain neural network classification,” *Neural Networks*, vol. 15, no. 2, pp. 237–246, Mar. 2002, doi: 10.1016/S0893-6080(01)00127-7.

[98] Y. Freund and R. E. Schapire, “Large margin classification using the perceptron algorithm,” *Mach. Learn.*, vol. 37, no. 3, pp. 277–296, 1999, doi: 10.1023/A:1007662407062.

[99] J. Schmidhuber, “Annotated History of Modern AI and Deep Learning,” Dec. 2022, [Online]. Available: http://arxiv.org/abs/2212.11279

[100] M. Matsugu, K. Mori, Y. Mitari, and Y. Kaneda, “Subject independent facial expression recognition with robust face detection using a convolutional neural network,” *Neural Networks*, vol. 16, no. 5–6, pp. 555–559, Jun. 2003, doi: 10.1016/S0893-6080(03)00115-1.

[101] M. Bahri, R. Ashino, and R. Vaillancourt, “Convolution theorems for quaternion fourier transform: Properties and applications,” *Abstr. Appl. Anal.*, vol. 2013, pp. 1–10, 2013, doi: 10.1155/2013/162769.

[102] R. Yamashita, M. Nishio, R. K. G. Do, and K. Togashi, “Convolutional neural networks: an overview and application in radiology,” *Insights Imaging*, vol. 9, no. 4, pp. 611–629, Aug. 2018, doi: 10.1007/s13244-018-0639-9.

[103] K. Fukushima, “Cognitron: A self-organizing multilayered neural network,” *Biol. Cybern.*, vol. 20, no. 3–4, pp. 121–136, 1975, doi: 10.1007/BF00342633.

[104] P. Ramachandran, B. Zoph, and Q. V. Le, “Searching for activation functions,” *6th Int. Conf. Learn. Represent. ICLR 2018 - Work. Track Proc.*, Oct. 2018.

[105] C. Nwankpa, W. Ijomah, A. Gachagan, and S. Marshall, “Activation Functions: Comparison of trends in Practice and Research for Deep Learning,” Nov. 2018, [Online]. Available: http://arxiv.org/abs/1811.03378

[106] J. Han and C. Moraga, “The influence of the sigmoid function parameters on the speed of backpropagation learning,” in *Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)*, 1995, pp. 195–201. doi: 10.1007/3-540-59497-3\_175.

[107] S. Ioffe and C. Szegedy, “Batch normalization: Accelerating deep network training by reducing internal covariate shift,” *32nd Int. Conf. Mach. Learn. ICML 2015*, vol. 1, pp. 448–456, Feb. 2015.

[108] N. Srivastava, G. Hinton, A. Krizhevsky, I. Sutskever, and R. Salakhutdinov, “Dropout: A simple way to prevent neural networks from overfitting,” *J. Mach. Learn. Res.*, vol. 15, no. 1, pp. 1929–1958, 2014.

[109] G. Ghiasi, T. Y. Lin, and Q. V. Le, “Dropblock: A regularization method for convolutional networks,” *Adv. Neural Inf. Process. Syst.*, vol. 2018-Decem, pp. 10727–10737, Oct. 2018.

[110] S. Xie, R. Girshick, P. Dollár, Z. Tu, and K. He, “Aggregated residual transformations for deep neural networks,” in *Proceedings - 30th IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2017*, IEEE, Jul. 2017, pp. 5987–5995. doi: 10.1109/CVPR.2017.634.

[111] M. Tan and Q. V. Le, “EfficientNetV2: Smaller Models and Faster Training,” *Proc. Mach. Learn. Res.*, vol. 139, pp. 10096–10106, Apr. 2021.

[112] A. Dosovitskiy *et al.*, “an Image Is Worth 16X16 Words: Transformers for Image Recognition At Scale,” *ICLR 2021 - 9th Int. Conf. Learn. Represent.*, Oct. 2021.

[113] Z. Liu, H. Mao, C. Y. Wu, C. Feichtenhofer, T. Darrell, and S. Xie, “A ConvNet for the 2020s,” in *Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, IEEE, Jun. 2022, pp. 11966–11976. doi: 10.1109/CVPR52688.2022.01167.

[114] S. Zagoruyko and N. Komodakis, “Wide Residual Networks,” *Br. Mach. Vis. Conf. 2016, BMVC 2016*, vol. 2016-Septe, pp. 87.1-87.12, May 2016, doi: 10.5244/C.30.87.

[115] M. Lukasik, S. Bhojanapalli, A. K. Menon, and S. Kumar, “Does label smoothing mitigate label noise?,” *37th Int. Conf. Mach. Learn. ICML 2020*, vol. PartF16814, pp. 6404–6414, Mar. 2020.

[116] W.-T. Pan, “A new Fruit Fly Optimization Algorithm: Taking the financial distress model as an example,” *Knowledge-Based Syst.*, vol. 26, pp. 69–74, Feb. 2012, doi: 10.1016/j.knosys.2011.07.001.

[117] S. Mirjalili, S. M. Mirjalili, and A. Lewis, “Grey Wolf Optimizer,” *Adv. Eng. Softw.*, vol. 69, pp. 46–61, Mar. 2014, doi: 10.1016/j.advengsoft.2013.12.007.

[118] S. Mirjalili, “The Ant Lion Optimizer,” *Adv. Eng. Softw.*, vol. 83, pp. 80–98, May 2015, doi: 10.1016/j.advengsoft.2015.01.010.

[119] S. Mirjalili, “Moth-flame optimization algorithm: A novel nature-inspired heuristic paradigm,” *Knowledge-Based Syst.*, vol. 89, pp. 228–249, Nov. 2015, doi: 10.1016/j.knosys.2015.07.006.

[120] R. Venkata Rao, “Jaya: A simple and new optimization algorithm for solving constrained and unconstrained optimization problems,” *Int. J. Ind. Eng. Comput.*, pp. 19–34, 2016, doi: 10.5267/j.ijiec.2015.8.004.

[121] S. Mirjalili and A. Lewis, “The Whale Optimization Algorithm,” *Adv. Eng. Softw.*, vol. 95, pp. 51–67, May 2016, doi: 10.1016/j.advengsoft.2016.01.008.

[122] S. Mirjalili, “Dragonfly algorithm: a new meta-heuristic optimization technique for solving single-objective, discrete, and multi-objective problems,” *Neural Comput. Appl.*, vol. 27, no. 4, pp. 1053–1073, May 2016, doi: 10.1007/s00521-015-1920-1.

[123] S. Mirjalili, A. H. Gandomi, S. Z. Mirjalili, S. Saremi, H. Faris, and S. M. Mirjalili, “Salp Swarm Algorithm: A bio-inspired optimizer for engineering design problems,” *Adv. Eng. Softw.*, vol. 114, pp. 163–191, Dec. 2017, doi: 10.1016/j.advengsoft.2017.07.002.

[124] M. Bakhshipour, M. Jabbari Ghadi, and F. Namdari, “Swarm robotics search & rescue: A novel artificial intelligence-inspired optimization approach,” *Appl. Soft Comput.*, vol. 57, pp. 708–726, Aug. 2017, doi: 10.1016/j.asoc.2017.02.028.

[125] R. Masadeh, B. A., and A. Sharieh, “Sea Lion Optimization Algorithm,” *Int. J. Adv. Comput. Sci. Appl.*, vol. 10, no. 5, 2019, doi: 10.14569/IJACSA.2019.0100548.

[126] H. Yapici and N. Cetinkaya, “A new meta-heuristic optimizer: Pathfinder algorithm,” *Appl. Soft Comput.*, vol. 78, pp. 545–568, May 2019, doi: 10.1016/j.asoc.2019.03.012.

[127] A. A. Heidari, S. Mirjalili, H. Faris, I. Aljarah, M. Mafarja, and H. Chen, “Harris hawks optimization: Algorithm and applications,” *Futur. Gener. Comput. Syst.*, vol. 97, pp. 849–872, Aug. 2019, doi: 10.1016/j.future.2019.02.028.

[128] L. Abualigah, D. Yousri, M. Abd Elaziz, A. A. Ewees, M. A. A. Al-qaness, and A. H. Gandomi, “Aquila Optimizer: A novel meta-heuristic optimization algorithm,” *Comput. Ind. Eng.*, vol. 157, p. 107250, Jul. 2021, doi: 10.1016/j.cie.2021.107250.

[129] O. O. Obadina, M. A. Thaha, K. Althoefer, and M. H. Shaheed, “Dynamic characterization of a master–slave robotic manipulator using a hybrid grey wolf–whale optimization algorithm,” *J. Vib. Control*, vol. 28, no. 15–16, pp. 1992–2003, Aug. 2022, doi: 10.1177/10775463211003402.

[130] A. Faramarzi, M. Heidarinejad, S. Mirjalili, and A. H. Gandomi, “Marine Predators Algorithm: A nature-inspired metaheuristic,” *Expert Syst. Appl.*, vol. 152, p. 113377, Aug. 2020, doi: 10.1016/j.eswa.2020.113377.

[131] F. A. Hashim, E. H. Houssein, K. Hussain, M. S. Mabrouk, and W. Al-Atabany, “Honey Badger Algorithm: New metaheuristic algorithm for solving optimization problems,” *Math. Comput. Simul.*, vol. 192, pp. 84–110, Feb. 2022, doi: 10.1016/j.matcom.2021.08.013.

[132] A. Seyyedabbasi and F. Kiani, “Sand Cat swarm optimization: a nature-inspired algorithm to solve global optimization problems,” *Eng. Comput.*, vol. 39, no. 4, pp. 2627–2651, Aug. 2023, doi: 10.1007/s00366-022-01604-x.

[133] L. Xie, T. Han, H. Zhou, Z.-R. Zhang, B. Han, and A. Tang, “Tuna Swarm Optimization: A Novel Swarm-Based Metaheuristic Algorithm for Global Optimization,” *Comput. Intell. Neurosci.*, vol. 2021, pp. 1–22, Oct. 2021, doi: 10.1155/2021/9210050.

[134] L. Wang, Q. Cao, Z. Zhang, S. Mirjalili, and W. Zhao, “Artificial rabbits optimization: A new bio-inspired meta-heuristic algorithm for solving engineering optimization problems,” *Eng. Appl. Artif. Intell.*, vol. 114, p. 105082, Sep. 2022, doi: 10.1016/j.engappai.2022.105082.

# واژه نامه

|  |  |
| --- | --- |
|  |  |
| 5-Fold Cross Validation | ارزیابی 5-قسمتی |
| Accuracy | دقت |
| Agent | نماینده |
| Apple Foliar Disease Network | بیماری‌های قارچی |
| Apple Scab | زخم سیب |
| Artificial Intelligence | هوش مصنوعی |
| Artificial Neural Network | شبکه عصبی مصنوعی |
| Automatic | خودکار |
| Average Pooling | ادغام میانگین |
| Backpropagation Algorithm | الگوریتم پس‌انتشار |
| Batch normalization | نرمالسازی دسته |
| Batch Size | سایز دسته |
| Bayes Classifier | طبقه‌بند بایزی |
| Bias | بایاس |
| Binary Classification | طبقه‌بندی دوتایی |
| Black Rot | لکه‌ی سیاه |
| Bottleneck | گلوگاه |
| Cardinality | کاردینالیته |
| Cedar Apple Rust | زنگار سیب-سرو |
| Chloroplast | کلروپلاست |
| Class | طبقه |
| Classification | طبقه‌بندی |
| Class-weight | طبقه-وزن |
| Cluster-based Region Reweighting | وزن‌بندی ناحیه ای خوشه ای |
| Complex | پیچیده |
| Compound Coefficient | ضریب ترکیب |
| Computational Intelligence | هوش محاسباتی |
| Computer Science | علوم کامپیوتر |
| Convolutional Neural Network | شبکه‌ی عصبی پیچشی |
| Cuticle | کوتیکول |
| Cutmix | برش-ترکیب |
| Cutout | برش تصویر |
| Data-Augmentation | داده‌افزایی‌ |
| Dataset | مجموعه‌داده |
| Deep Learning | یادگیری عمیق |
| Deep Neural Network | شبکه عصبی عمیق |
| Defoliation | برگ‌زدایی |
| Pollinators | گرده‌افشانی |
| Distribution | پراکندگی |
| Drone | پهپاد |
| Drop Block | بلوک برون‌انداز |
| Dropout | برون‌انداز |
| Dying | مرگ |
| Ensemble | ترکیب مدل |
| Epidermis | اپی‌درم |
| Epoch | دوره |
| False Negative | منفی غلط |
| False Positive | مثبت غلط |
| Feature Map | نقشه ویژگی |
| Features | ویژگی‌ها |
| Fine-Grained Visual Categorization | طبقه‌بندی بصری دقیق |
| Fine-Tune | تنظیم دقیق |
| Fitness Function | تابع تناسب |
| Flip | آیینه کردن |
| Float | اعشاری |
| Food security | امنیت غذایی |
| Frame | قاب |
| Frog Eye Leaf Spot | لکه برگ چشم غورباقه‌ای |
| Fully Connected Layer | تمام متصل |
| Gas Chromatography | رنگ‌شناسی گازی |
| Gaussian Blur | تارسازی گوسی |
| Gaussian error Linear Unit | واحد خطی خطای گوسی |
| Generative Adaptive Neural Network | شبکه‌ی مولد تخاصمی |
| Genetic Algorithm | الگوریتم ژنتیک |
| Global Average Pooling | ادغام میانگین کلی |
| Hierarchial | سلسله مراتبی |
| Hyperparameter | ابرپارامتر |
| Image Analysis | بررسی تصویر |
| Image Enhancement | بهبود تصویر |
| Image Preprocessing | پیش‌پردازش تصویر |
| Image Processing | پردازش تصویر |
| Image Sensors | حسگر‌های تصویر |
| Image Transformation | تبدیل تصویر |
| Inverted Bottleneck | گللوگاه معکوس |
| Kernel | کرنل |
| K-means clustering | خوشه‌بندی ک-میانگین |
| Label | برچسب |
| Lambda | لامبدا |
| Learning From Scratch | یادگیری از ابتدا |
| Learning Rate | نرخ یادگیری |
| Learning Rate Scheduler | تغییردهنده نرخ یادگیری |
| Long Short-Term Memory | حافظه بلند کوتاه‌مدت |
| Loss | هزینه |
| Loss Function | تابع هزینه |
| Machine Learning | یادگیری ماشین |
| Max Pooling | ادغام بیشینه |
| Median Blur | تارسازی میانه‌ای |
| Mega Flop | مگافلاپ |
| Mesophyll | مزوفیل |
| Microorganisms | ریزجانداران |
| Mixup | مخلوط‌کردن |
| Movement Strategies | استراتژی‌های حرکتی |
| Multi-Class Classification | طبقه‌بندی چند طبقه |
| Multi-Label Classification | طبقه‌بندی چند برچسبه |
| Multiple Disease | چندین بیماری |
| Natural Laguage Proccessing | پردازش زبان طبیعی |
| Neighborhood Structure | ساختار همسایگی |
| Node | گره |
| Northern Leaf Blight | لهیدگی شمالی برگ |
| Open-Source | متن - باز |
| Optimizer | بهینه‌ساز |
| Overfitting | بیش‌برازش |
| Padding | پَدینگ |
| Patch | بخش |
| Perceptron | پرسپترون |
| Pesticide | آفت‌کش |
| Photosynthesis | فتوسنتز |
| Polymerase Chain Reaction | واکنش زنجیره‌ای پلیمراز |
| Pooling | ادغام |
| Powdery Mildew | کپک پودری |
| Progressive Training | آموزش پیشرفتی |
| RAM | حافظه موقت |
| Raw Data | داده‌ی خام |
| Reccurrent Neural Networks | شبکه‌های عصبی بازگشتی |
| Rectified Linear Unit | واحد یکسوساز خطی |
| Regularization | تنوع‌سازی |
| Reinforcement Learning | یادگیری تقویتی |
| RGB code | کد قرمز – سبز – آبی |
| Rust | زنگار |
| Semi-Supervised Learning | یادگیری نیمه نظارت |
| Shifted Windows Transformer | مبدل پنجره‌ای جابه‌جا شده |
| Shift-Scale | انتقال مرکز |
| Sigmoid | سیگموید |
| Softmax | بیشینه هموار |
| Spores | اسپور |
| SSD | حافظه جامد |
| Step Activation Function | تابع فعال‌سازی پله‌ای |
| Supervised Learning | یادگیری با نظارت |
| Swarm Intelligence | هوش ازحامی |
| Swarm-based Optimization Algorithm | الگوریتم بهینه‌سازی ازدحامی |
| Swish | سوویش |
| Tensor | تنسور |
| Test | آزمون |
| TF Record | کد رکوردی |
| Threshold | آستانه |
| Thresholidng | آستانه‌گذاری |
| Training with Loss Reweighting | آموزش با وزن بندی اختلاف خطا |
| Transformers | مبدل‌ها |
| Triangular Learning Rate Scheduler | تنظیم‌کننده نرخ یادگیری مثلثی |
| True Positive | مثبت صحیح |
| TrueNegative | مثبت غلط |
| Unsupervised Learning | یادگیری بدون نظارت |
| Update | به‌روزرسانی |
| User Interface | رابط کاربری |
| Validation | اعتبارسنجی |
| Vanishing Gradient | محوشدگی گرادیان |
| VRAM | حافظه گرافیکی |
| Weakly-Supervised Training | آموزش نظارتی ضعیف |
| Weight Decay | کاهش وزن |
| Weighted Feature Integration | اعمال ویژگی‌های وزن بندی شده |

**Abstract:**

Plant diseases severrly affect agricultural products and cause their destruction, it is necessary to identify and deal with as soon as possible. One of the important organs of a plant is its leaf; which is crucial for it’s survival and growth. The traditional methods of identifying plant leaf diseases include specialists visiting the farm or orchard and observing a large number of plant leaves by eye or sending samples of the leaf to the laboratory. These methods are time-consuming and often cost a lot. With the remarkable advancements in Image Processing and Deep Learning, it is possible to design systems capable of automatically classifying leaf diseases and diagnosing them with high accuracy. In this research, ten Convolutional Neural Networks and Image Transformers are investigated to find the best performance. The dataset used to train, validate and test the models is Plant Pathology 2021, which contains 18632 images of apple leaves in 12 classes. The classification method used in this research is Multi-Label Claassification. The structures of models have been modified and Dropout, DropBlock and a 6 layer fully-connected layer have been added to the networks. Best performance was achieved by Resnext5032x4D, SwinV2T and SwinV2S with an accuracy of 92.62%, 92.22% and 92.51% and average F-1 scores of 95.31%, 94.77% and 94.75%, respectively. Another test has been carried out in which the thresholds of the models are changed by Swarm Optimization Algorithms. 19 of these algorithms have been chosen which had two parameters. After finding the optimized thresholds, an increase in accuracy in all the networks and an increase in F-1 score in most the models were observed. The best results after the optimization are for the models Resnext5032x4D, SwinV2T and SwinV2S with accuracies of 93.32%,92.18% and 93% and F-1 scores of 95.4%, 94.96% and 94.88%, respectively. Also the optimization algorithms were ranked according to their performance on the networks and the best ones were chosen. The training methods and newly introduced models have better performance on the dataset than previous reasearches.

**Keywords:** Deep Learning, Convolutional Neural Networks, Image Transformers, Swarm Optimization Algorithms, DropBlock, Precision Agriculture, Image Processing, Plant Pathology Systems

|  |  |  |
| --- | --- | --- |
|  | P1969#yIS1 |  |
| University of Tehran  Faculty of New Sciences and Technologies  Department of Interdisciplinary Technology | | |
| Title:  Classification of Plant Leaves Diseases Based on Convolutional Neural Network Learning | | | |
| By:  Soroush Toutounchian | | | |
| Supervisor:  Dr. Alireza Rezaee  Advisor: Dr. Farshid Hajati | | | |
|  | | | |
| A Thesis Submitted to the Graduate Office in Fulfillment  of Requirements for the Degree of Master of Science  in Mechatronics Engineering | | | |
| September 2023 | | | |

1. Multi-robot [↑](#footnote-ref-1)
2. Cooperative [↑](#footnote-ref-2)
3. Trajectory tracking [↑](#footnote-ref-3)
4. Network [↑](#footnote-ref-4)
5. Uncertainty [↑](#footnote-ref-5)
6. Legged Mobile Robots [↑](#footnote-ref-6)
7. Wheeled Mobile Robots [↑](#footnote-ref-7)
8. RPM [↑](#footnote-ref-8)
9. Multi-Robot system [↑](#footnote-ref-9)
10. Trajectory tracking [↑](#footnote-ref-10)
11. Framework [↑](#footnote-ref-11)
12. Formation [↑](#footnote-ref-12)
13. Consensus [↑](#footnote-ref-13)
14. Asymptotically [↑](#footnote-ref-14)
15. Uncertainty [↑](#footnote-ref-15)
16. Distributed control systems [↑](#footnote-ref-16)
17. Leader [↑](#footnote-ref-17)
18. Deterministic learning theory [↑](#footnote-ref-18)
19. Constraints [↑](#footnote-ref-19)
20. State constraints [↑](#footnote-ref-20)
21. Model predictive control [↑](#footnote-ref-21)
22. Error transformation function [↑](#footnote-ref-22)
23. Barrier Lyapunov function [↑](#footnote-ref-23)
24. Uncertain [↑](#footnote-ref-24)
25. Radial basis function [↑](#footnote-ref-25)
26. Equilibrium point [↑](#footnote-ref-26)
27. Aleksandr Lyapunov [↑](#footnote-ref-27)
28. Markov chains [↑](#footnote-ref-28)
29. State space [↑](#footnote-ref-29)
30. Conservation laws [↑](#footnote-ref-30)
31. Continuous [↑](#footnote-ref-31)
32. Homeomorphism [↑](#footnote-ref-32)
33. 1 Le Cunn [↑](#footnote-ref-33)
34. 2 Supervised Learning [↑](#footnote-ref-34)
35. 3 Unsupervised Learning [↑](#footnote-ref-35)
36. 4 Semi-Supervised Learning [↑](#footnote-ref-36)
37. 5 Reinforcement Learning [↑](#footnote-ref-37)
38. 1 Float [↑](#footnote-ref-38)
39. 2 Perceptron [↑](#footnote-ref-39)
40. 3 Bias [↑](#footnote-ref-40)
41. 1 Step Activation Function [↑](#footnote-ref-41)
42. 2 McCulloch-Pitts [↑](#footnote-ref-42)
43. 3 Backpropagation Algorithm [↑](#footnote-ref-43)
44. 1 Pooling [↑](#footnote-ref-44)
45. 2 Batch Normalization Layer [↑](#footnote-ref-45)
46. 3 Fully-Connected Layer [↑](#footnote-ref-46)
47. 4 Kernel [↑](#footnote-ref-47)
48. 1 Feature Map [↑](#footnote-ref-48)
49. 1 Rectified Linear Unit (ReLU) [↑](#footnote-ref-49)
50. 2 Dying [↑](#footnote-ref-50)
51. 3 Sigmoid [↑](#footnote-ref-51)
52. 4 Vanishinig Gradient [↑](#footnote-ref-52)
53. 1 Swish [↑](#footnote-ref-53)
54. 1 Padding [↑](#footnote-ref-54)
55. 1 Max Pooling [↑](#footnote-ref-55)
56. 2 Average Pooling [↑](#footnote-ref-56)
57. 3 Global Average Pooling [↑](#footnote-ref-57)
58. 4 Dropout [↑](#footnote-ref-58)
59. 1 Overfitting [↑](#footnote-ref-59)
60. 2 Drop Block [↑](#footnote-ref-60)
61. 3 Fully Connected Layer [↑](#footnote-ref-61)
62. 1 Binary Classification [↑](#footnote-ref-62)
63. 1 Cardinality [↑](#footnote-ref-63)
64. 1 Compound Coefficient [↑](#footnote-ref-64)
65. 1 Regularization [↑](#footnote-ref-65)
66. 2 Progressive Training [↑](#footnote-ref-66)
67. 1 Natural Laguage Proccessing [↑](#footnote-ref-67)
68. 2 Transformers [↑](#footnote-ref-68)
69. 3 Patch [↑](#footnote-ref-69)
70. 4 Shifted Windows Transformer [↑](#footnote-ref-70)
71. 5 Hierarchial [↑](#footnote-ref-71)
72. 1 Bottleneck [↑](#footnote-ref-72)
73. 2 Mega Flop [↑](#footnote-ref-73)
74. 1 Reccurrent Neural Networks [↑](#footnote-ref-74)
75. 1 Mixup [↑](#footnote-ref-75)
76. 2 Inverted Bottleneck [↑](#footnote-ref-76)
77. 3 Gaussian error Linear Unit (GeLU) [↑](#footnote-ref-77)
78. 1 ImageNet Large Scale Visual Recognition Challenge [↑](#footnote-ref-78)
79. 1 Fine-Tune [↑](#footnote-ref-79)
80. 2 Swarm-based Optimization Algorithm [↑](#footnote-ref-80)
81. 3 Computational Intelligence [↑](#footnote-ref-81)
82. 4 Swarm Intelligence [↑](#footnote-ref-82)
83. 1 Agent [↑](#footnote-ref-83)
84. 2 Neighborhood Structure [↑](#footnote-ref-84)
85. 3 Movement Strategies [↑](#footnote-ref-85)
86. 4 RAM [↑](#footnote-ref-86)
87. 5 Tensor [↑](#footnote-ref-87)
88. 1 Gudio van Rossum [↑](#footnote-ref-88)
89. 2 True Positive [↑](#footnote-ref-89)
90. 3 TrueNegative [↑](#footnote-ref-90)
91. 4 False Positive [↑](#footnote-ref-91)
92. 5 False Negative [↑](#footnote-ref-92)
93. 1 VRAM [↑](#footnote-ref-93)
94. 2 SSD [↑](#footnote-ref-94)
95. 3 Open-Source [↑](#footnote-ref-95)
96. 1 Label Smoothing [↑](#footnote-ref-96)
97. 1 Genetic Algorithm [↑](#footnote-ref-97)
98. 2 Fitness Function [↑](#footnote-ref-98)
99. 1 Triangular Learning Rate Scheduler [↑](#footnote-ref-99)
100. 2 Median Blur [↑](#footnote-ref-100)
101. 3 Gaussian Blur [↑](#footnote-ref-101)
102. 1 Google Colab [↑](#footnote-ref-102)
103. 2 Kaggle Notebook [↑](#footnote-ref-103)
104. 1 Generative Adaptive Neural Network [↑](#footnote-ref-104)
105. 2 Jetson Nano [↑](#footnote-ref-105)