توضیح مسئله :

در این مسئله باید برای دسته بندی داده های دیتاست CFAR10 بهترین معماری شبکه عصبی ممکن را پیدا کنیم اما تعداد حالات زیاد است و باید از الگوریتم تکاملی استفاده کنیم تا تعداد حالات کنترل شود و به جواب های قابل قبولی برسیم. استفاده از الگوریتم تکاملی به این صورت خواهد بود که از شبکه های استخراج ویژگی نظیر Vgg_11و Resnet_34 و Resnet_18 استفاده میکنیم و تعداد لایه های شبکه های عصبی مان 0 یا 1 و یا 2 لایه خواهند بود همچنین تعداد نورون ها در هر لایه مقداری برابر 10 یا 20 یا 30 خواهد داشت.
در این مسئله پارامتر های ما وزن ها و بایاس ها هستند که با استفاده از کتابخانه های آماده آموزش داده میشوند و ابر پارامترهای ما که آموزش داده نمیشوند باید با تکامل مقادیرشان را به روزرسانی کرد تعداد لایه های مخفی شبکه و تعداد نورون های هر لایه و تابع فعال سازی در هر لایه مخفی و نوع شبکه استخراج کننده ویژگی هستند.

هدف ما بهبود ساختار شبکه عصبی مورد استفاده برای مسئله دسته بندی داده های CFAR10 است.

نحوه پیاده سازی :

ابتدا یک جمعیت اولیه از شبکه های عصبی می سازیم ، کروموزوم های ما همون معماری های شبکه های عصبی هستند. جمعیت اولیه کاملا بر اساس رندوم ساخته میشوند و سپس نسل اولیه به تابع ارزیابی میرود تا هر کدام از افراد این نسل اولیه امتیاز دهی بشوند تا فرایند تولید نسل های بعد آغاز شود.

در تابع ارزیابی جمعیت اولیه را آموزش میدهیم با داده های CFAR10 و یک نمره دهی به هر معماری شبکه عصبی موجود در جمعیت میکنیم سپس از بین بهترین های هر نسل تعدادی از آن ها را ترکیب میکنیم و جهش میدهیم تا شانس ایجاد شبکه های بهتری برای نسل بعد وجود داشته باشد .

نحوه انتخاب افراد هر نسل برای استفاده شدن در تولید نسل بعد :

این انتخاب بر اساس یک مقدار دستی به نام درصد نخبگان هر نسل معین میشود که مشخص میکند چه تعداد از برترین افراد هر نسل امکان انتقال به نسل بعد را خواهند داشت.

تعیین معیار برترین افراد هر نسل با مرتب کردن افراد هر نسل به ترتیب بیشترین fitness_score به کمترین مقدار آن بین افراد هر نسل است و سپس افراد نخبه از بین برترین افراد نسل انتخاب خواهند شد.

در انتخاب اینکه چه تعداد از کروموزوم های یک نسل امکان رفتن به نسل بعد را دارند در این الگوریتم تعدادی از افراد نسل بعد افراد نخبه این نسل هستند که مستقیم به نسل بعد خواهند رفت و بقیه افراد نسل بعد به صورت رندوم از بین تمام افراد نسل فعلی انتخاب میشوند و روی آن ها ترکیب و جهش اعمال میشود و سپس به نسل بعد اضافه میشوند، و نسل جدید جایگزین نسل فعلی می شود .

تابع فیتنس :

میزان فیتنس هر کروموزوم به این صورت تعیین میشود که هر معماری 25 بار آموزش داده میشود و 5 بار در کل تست میشود و مقدار هر بار accuracy که بعد از هر تست به دست می آید باهم جمع میشوند و نتیجه تقسیم بر 5 (تعداد تست شبکه) میشوند که در اصل میانگین کل accuracy های به دست آمده است به عنوان مقدار فیتنس آن معماری انتخاب میشود.

تابع کراس اور :

دو والد که به صورت رندوم از بین جمعیت نسل فعلی انتخاب شدند به تابع کراس اور یا ترکیب داده میشوند.

در این تابع یک مقدار به اسم crossover_point تعیین میشود که یک عدد است و در اصل به شماره یک لایه مشترک بین دو والد انتخاب شده اشاره میکند.

سپس تعداد نورون های موجود در لایه تعیین شده از والد اول با تعداد نورون های موجود در همان لایه از والد دوم جابه جا میشوند.
همینطور مقدار activation_function این لایه از هر دو والد هم با هم جا به جا میشوند.

در نهایت از دو والد دو فرزند تولید میشوند و به صورت رندوم تابع استخراج کننده ویژگی هر کدام از این دو فرزند از بین دو تابع استخراج کننده دو والد انتخاب میشوند و درنهایت شیئ ساخته شده این دو فرزند برگشت داده میشوند.

تابع جهش :

در این تابع ما براساس یک مقدار نرخ جهش که به صورت دستی به الگوریتم داده بودیم و یک عدد که به صورت رندوم میسازیم تصمیم میگیریم که روی لایه های یک معماری شبکه عصبی جهش بزنیم یا نه .

این عملیات به تعداد لایه های آن معماری انجام میشود و برای هر لایه سه بار عدد رندومی تولید میشود که دفعه اول اگر کوچک تر از مقدار نرخ جهش تعیین شده از سوی ما بود جهش روی تعداد نورون های آن لایه انجام میشود. دفعه دوم اگر کوچک تر از مقدار نرخ جهش بود مقدار تابع فعال سازی آن لایه جهش انجام میشود. دفعه سوم کوچک تر از مقدار نرخ جهش بود جهش روی نوع شبکه استخراج کنندگی آن شبکه عصبی صورت میگیرد.

در تمامی انواع جهش ، منظور از جهش انتخاب دوباره رندوم یک مقدار از مقادیر از پیش تعریف شده برای هر کدام از ابر پارامتر ها است و نه اضافه یا کم کردن یک مقدار دلخواه به ابر پارامتر ها.

بنابراین در هر جهش امکان دارد تعداد نورون های یک لایه بیشتر یا کمتر بشود ولی بر اساس مقادیر از پیش تعیین شده تغییر میکند.

نحوه تعیین برترین معماری :

در هر نسل بهترین شبکه آن نسل را ذخیره میکنیم تا در نهایت پس از ساخت 10 نسل و ارزیابی تمام 10 کروموزوم آن نسل بتوانیم در نهایت برترین کروموزوم های 10 نسل را که ذخیره کرده بودیم مقایسه کنیم و بهترین آن ها را به عنوان بهترین معماری معرفی کنیم .

پس از چندین بار اجرای کد بهترین خروجی متعلق به معماری با مشخصات زیر بود با accuracy = 62% :

Best MLP Neural Network Architecture:

Number of Layers: 2

Number of Neurons per Layer: [20, 10]

Activation Function: ['relu', 'sigmoid']

Feature Extraction Network: Vgg_11

Accuracy: 0.6228999972343445