# مقدمه:

تکنیک‌های یادگیری تقویتی در کاربردهایی که به یادگیری بدون نظارت نیاز داریم کمک می‌کند و ثابت شده است که این تکنیک‌ها زمانی مؤثرتر عمل می‌کنند که برای تقریب‌گرها از توابع غیرخطی استفاده شود. به دلیل اهمیت یادگیری تقویتی برای حل مسائل، سال‌های اخیر، بازار تولید الگوریتم‌های جدید یادگیری تقویتی بسیار داغ شده است. هر جا در مورد الگوریتم و روش‌های جدید بحث می‌شود، تنها تولید الگوریتمی که به کار ما پاسخ دهد، پاسخگوی نیاز ما نیست، بنابراین نیاز داریم تا روش‌هایی نیز برای افزایش سرعت اجرای این الگوریتم‌ها نیز بیابیم. الگوریتم‌های یادگیری تقویتی، الگوریتم‌های پیچیده‌ای هستند که اگرچه پاسخ بهتری به ما می‌دهند اما از آن طرف هم به محاسبات سنگینی نیاز دارند.

برای اجرای سریع‌تر این الگوریتم‌ها، یکی از حوزه‌هایی که به سرعت در حال استقبال است، ورود به الگوریتم‌هایی است که این محاسبات را در بستر سیستم‌های توزیع شده انجام دهند. در این راستا، الگوریتم‌های تکاملی(ES) یکی از الگوریتم‌هایی است که نشان‌داده‌است می‌تواند با الگوریتم‌های تقویتی رقابت کند در حالی که همچنان فاصله زیادی تا آن‌ها دارد. الگوریتم‌های تکاملی در مقایسه با الگوریتم‌های تقویتی، الگوریتم‌هایی هستند که بهینه‌سازی را به صورت مستقیم و بدون محاسبه مشتق، حل می‌کنند؛ به عبارتی یا اصلاً از گرادیان استفاده نمی‌کنند (مانند الگوریتم ژنتیک) یا گرادیان را با به صورت محدود و متفاوت مورد استفاده قرار می‌دهند.

برخی از مزیت‌های الگوریتم‌های تکاملی عبارتند از:

1. از آن‌جا که ارزیابی مدل‌ها به صورت پاداش و مجازات را ندارند مسائل بسیار بزرگ را هم می‌توانند حل کنند.
2. دیگر هزینه‌های محاسباتی رفت و برگشتی برای تنظیم موقعیت و ارزیابی تابع ارزش را ندارند.
3. این الگوریتم‌ها را به راحتی بر روی سیستم‌های توزیع شده قابل اجرا خواهند بود.

اما دو مورد از اشکال‌های این الگوریتم‌ها:

1. شدیداً به حجم داده‌ها وابسته هستند و باید به اندازه کافی حجم داده‌هایی که در حل مسئله مورد استفاده قرار می‌گیرد بزرگ باشد.
2. معمولاً کارآیی کم‌تری نسبت به الگوریتم‌های تقویتی دارند و اگر تغییری در سیاست مربوط به داده‌ها داشته باشیم، باید الگوریتم بر روی تمامی داده‌ها اجرا شود. به عنوان مثال، بوت استرپ از برآورد فعلی تابع مقدار در تفاوت زمانی.

اما با این حال، محاسبات کم‌تر در الگوریتم‌های تکاملی منجر می‌شود محاسبات بد آن را تا حد زیادی در نظر نگیریم. در این مقاله قرار است نشان داده شود که با استفاده از یک راهبرد جدید و توزیع شده، می‌توان کارآیی الگوریتم‌های تکاملی را افزایش داد.

# الگوریتم تکاملی توزیع شده

به طور کلی، موازی‌سازی توابعی که بر پایه محاسبه گرادیان می‌باشند بین ماشین‌های مختلف، به شکلی که در برابر خطا مقاوم باشند، کار دشواری است اما الگوریتم‌های تکاملی این مشکل را ندارند زیرا نیازی به محاسبه بر پایه گرادیان نیستند. برخلاف توابعی که برپایه گرادیان کار می‌کنند و برای اجرا به یک معماری یکپارچه نیاز دارند، الگوریتم‌های جستجو مانند الگوریتم‌های تکاملی به گونه‌ای طراحی شده‌اند که موازی‌سازی یک پیاده‌سازی بدون دردسر باشد.

در الگوریتم‌های تکاملی فرضیه اصلی این است که از میان اعضای مختلفی که برای جستجو انتخاب شده‌اند، آن‌ها که عملکرد خوبی دارند برای تغییر انتخاب می‌شوند و دیگر اعضا، حذف می‌شوند. مهم نیست که اعضای جمعیت کجا آموزش دیده می‌شوند یا چگونه با گرادیان درستی نسبت به بقیه اعضا، درست عمل می‌کنند. سیاستی که برای انتخاب اعضای جمعیت در الگوریتم تکاملی انجام می‌شود، به راحتی بر روی سیستم‌های توزیع شده قابل اجرا خواهند بود.

## آموزش توزیع شده

ایده اصلی این الگوریتم به این صورت است که: یکی از سیستم‌ها به عنوان رهبر انتخاب می‌شود و دیگر سیستم‌ها نقش کارگر را دارند. انتخاب جمعیت برتر برای تغییر و آموزش بیش‌تر برعهده سیستم رهبر خواهد بود. وظیفه سیستم‌های کارگر محلی، انجام عمل محاسباتی اکتشاف، در هر دور اجرای الگوریتم می‌باشد. از آن‌جا که در این الگوریتم‌ها اکتشاف به صورت تصادفی است، بنابراین در مورد انجام کار تکراری توسط سیستم‌های مختلف، نگران نخواهیم بود. پس از انجام اکتشاف توسط سیستم‌های کارگر، این سیستم‌ها اکتشاف خود را تا حدی بهبود می‌بخشند و پاسخ خود را به گرادیان نزدیک می‌کنند و در نهایت برای رهبر ارسال می‌کنند. رهبر از میان جمعیت‌های مختلف، جمعیت برتر را انتخاب می‌کند. این چرخه آنقدر تکرار می‌شود که الگوریتم به یکی از این دو شرط برسد:

1. تعداد تکرارها به تعدادی برسد که از پیش تعیین شده است.
2. میزان کارآیی بیش‌تر از عددی شود که از پیش تعیین شده است.

پیش از این نیز روش‌های توزیع‌شده برای الگوریتم‌های تکاملی مطرح شده بودند که برپایه جابه‌جا کردن مدل‌ها میان سیستم‌ها هستند. اما همان‌طور که واضح است، جا‌به‌جایی مدل میان سیستم‌ها، باری سنگین‌تر از آن‌چه که بشود تصور کرد را بر روی شبکه ایجاد می‌کند و منجر به تأخیر خواهد شد. برای کاهش این پیچیدگی پیامی، به جای ارسال مدل‌های ایجاد شده می‌توان seed ای که از آن برای تولید مدل‌ها استفاده شده را برای دیگر سیستم‌ها ارسال کرد. نمونه الگوریتم تکاملی توزیع شده بر پایه ارسال seed در قسمت زیر نمایش داده شده است:

|  |
| --- |
| Algorithm 1: Distributed ES |
| **Input:** Leader, Workers, number of iterations i, environment e, noise matrix generator gen(), population size p  **Output:** Trained model M  Note: assume model is a function, its output is a fitness function, and we want to maximize the fitness function.  1 Each of Workers generates duplicated M from random  2 Leader generates  3 for i iterations:  4 Leader creates empty list  5 for each worker in Workers:  6 Leader sends to worker  7 worker updates  8 worker generates try seeds, list length p of random numbers  9 worker creates model fitness tuple  10 for j in range(p):  11  12  13 worker appends to  14 worker sends w.r.t. back to Leader  15 Leader appends from each worker to  16 w.r.t.  17 Leader updates  18 Leader returns |