شبکه‌های کامپیوتری پیشرفته
پاییز ۱۴۰۴

دکتر دولتی

پروژه چهارم

Distributed Machine Learning

۱ مقدمه

اهمیت و کارایی یادگیری ماشین در حل مسائل گوناگون و پیچیده‌ای که در دنیای واقعی با آن‌ها مواجه هستیم، بر هیچ‌کس پوشیده نیست. با وجود این قابلیت‌ها، موانع و چالش‌های متعددی همچنان بر سر راه گسترش و بهره‌برداری فراگیر از این فناوری وجود دارد. از جمله این چالش‌ها می‌توان به نیاز قابل توجه به حجم عظیمی از داده‌ها و منابع محاسباتی اشاره کرد. اکثر مدل‌های پیشرفته یادگیری ماشین، که معمولا مبتنی بر معماری شبکه‌های عصبی ژرف هستند، برای دستیابی به دقت بالا و تعمیم مناسب، مستلزم انجام محاسبات موازی گسترده می‌باشند.

برای تأمین این توان پردازشی، یکی از راهکارهای ممکن، افزایش ظرفیت و قدرت سخت‌افزاری سرورهایی است که فرایند آموزش مدل‌ها بر روی آن‌ها انجام می‌گیرد؛ با این حال، این افزایش تا حد معینی امکان‌پذیر است و پس از آن باید به دنبال رویکردهای جایگزین بود. یکی از این راهکارها، توزیع فرایند یادگیری میان چندین دستگاه یا گره محاسباتی است.

علاوه بر نیاز به توان پردازشی بالا و همان‌گونه که پیش‌تر اشاره شد، مدل‌های یادگیری ماشین به مقادیر قابل توجهی داده نیز وابسته‌اند. گردآوری و فراهم‌سازی این داده‌ها غالباً با چالش‌های گوناگونی همراه است که یکی از مهم‌ترین آن‌ها مسئله حفظ امنیت و محرمانگی داده‌ها است. داده‌های متعلق به سازمان‌ها عموماً از ارزش راهبردی و اقتصادی بالایی برخوردارند و افشای آن‌ها می‌تواند تبعات جدی و بعضاً جبران‌ناپذیری برای آن سازمان‌ها به همراه داشته باشد. از همین رو، معمولاً این داده‌ها برای اهداف پژوهشی یا آموزشی در اختیار سایر نهادها و سازمان‌ها قرار نمی‌گیرند.

یکی از رویکردهای نوین که به منظور رفع هم‌زمان دو چالش مذکور مورد استفاده قرار می‌گیرد، روش یادگیری فدرال است. در این شیوه، فرایند آموزش مدل میان چندین پردازشگر مستقل توزیع می‌شود. هر یک از این پردازشگرها دارای ساختار کلی و معماری یکسانی از مدل مورد نظر برای آموزش هستند، اما به تمامی داده‌ها دسترسی ندارند و صرفاً از بخش اختصاص‌یافته به خود استفاده می‌کنند. در جریان فرایند یادگیری، هر پردازشگر با اجرای الگوریتم کاهش گرادیان در هر تکرار، وزن‌های شبکه‌ی محلی خود را به‌روزرسانی می‌کند. سپس در بازه‌های زمانی معین (پس

از یک یا چند مرحله یادگیری)، وزن‌های به‌دست‌آمده را با سایر پردازشگرها به اشتراک می‌گذارد. هر پردازشگر پس از دریافت وزن‌های به‌روزرسانی‌شده از سایرین، میانگین وزن‌های دریافتی را همراه با وزن‌های مدل خود محاسبه کرده و مدل محلی خویش را بر اساس این میانگین به‌روزرسانی می‌کند. این فرایند موجب می‌شود که مدل نهایی بدون نیاز به تجمیع داده‌های خام در یک مکان مرکزی، به شکلی هماهنگ و هم‌زمان در میان چندین واحد محاسباتی آموزش ببیند، در حالی که محرمانگی داده‌ها نیز حفظ می‌شود.

همان‌گونه که قابل پیش‌بینی است، در چنین ساختاری بخشی از چالش‌های موجود در فرایند یادگیری ماشین تا حدی برطرف می‌شود، اما در عین حال، گلوگاه اصلی سیستم از سطح سرورها به سطح شبکه منتقل می‌گردد. تبادل مکرر وزن‌ها میان پردازشگرها منجر به افزایش حجم ترافیک شبکه و در نتیجه، کاهش سرعت و کارایی کلی فرایند یادگیری می‌شود. برای رفع این مشکل، روش‌های متعددی پیشنهاد شده است که یکی از مهم‌ترین آن‌ها بهره‌گیری از شبکه‌هایی با لایه داده برنامه‌پذیر است. این نوع شبکه‌ها که می‌توان آن‌ها را به‌منزله‌ی نسل بعدی شبکه‌های نرم‌افزارمحور در نظر گرفت، از سوییچ‌هایی استفاده می‌کنند که نه‌تنها وظایف معمول لایه داده را بر عهده دارند، بلکه خود نیز از قابلیت برنامه‌پذیری برخوردار بوده و به‌صورت نرم‌افزاری قابل پیکربندی و تغییر هستند. مزیت اصلی این ویژگی در آن است که امکان تعریف و استقرار پروتکل‌های جدید و اختصاصی با هزینه‌ای اندک فراهم شده و می‌توان بخشی از عملیات پردازشی مورد نیاز را نیز به این سوییچ‌های برنامه‌پذیر واگذار کرد. این امر ضمن کاهش بار پردازشی بر روی سرورها، موجب بهبود بهره‌وری شبکه و تسریع روند هماهنگی میان پردازشگرها در محیط‌های یادگیری توزیع‌شده می‌شود.

در این پروژه هدف ما طراحی و پیاده‌سازی یک پروتکل مبتنی بر شبکه‌های دارای لایه داده برنامه‌پذیر با استفاده از زبان P4 است که مسئولیت جمع‌آوری، نگهداری و در نهایت پخش وزن‌های مدل‌های یادگیری ماشین را بر عهده گرفته و باعث کاهش سربار تبادل درون‌شبکه‌ای می‌شود. با توجه به اینکه مقصود آموزشی این پروژه آشنایی شما با مفاهیم شبکه داده‌محور برنامه‌پذیر و زبان برنامه‌نویسی P4 است، مجموعه‌ای از مؤلفه‌ها در اختیار شما قرار می‌گیرد:

- یک کد پایتون که نقش پردازشگر را ایفا کرده و مراحل آموزش مدل را اجرا می‌کند.
- قالب و چارچوب کد زبان P4 برای اجرا بر سوییچ‌ها.

شما باید ابتدا با بهره‌گیری از کدهای فراهم‌شده، پروتکل انتقال وزن‌ها را طراحی کنید؛ سپس منطق مربوط به جمع‌آوری و انتشار این وزن‌ها را در محیط P4 پیاده‌سازی نمایید. در مرحله بعد، با استفاده از عملگرها و ابزارهایی که P4 فراهم می‌آورد، داده‌های مختلف شبکه‌ای را استخراج کرده و این داده‌ها را در کنترل‌کننده مرکزی تجمیع، تحلیل و نمایش دهید. در نهایت، کد پردازشگر پایتون را متناسب با ساختار پروتکلی طراحی‌شده تنظیم کنید تا وزن‌ها را به سوییچ‌های میانی ارسال نماید و پس از آن بسته‌های حاوی وزن‌های جمع‌آوری‌شده از سوی سوییچ‌ها را دریافت و پردازش کند.

۲ محیط اجرایی

برای پیاده‌سازی این پروژه از یک محیط شبیه‌سازی شبکه‌ای مبتنی بر ابزارهای متن‌باز استفاده شد. زیرساخت آزمایشی با بهره‌گیری از شبیه‌ساز Mininet و مجموعه ابزار P4-Utils ایجاد گردید تا امکان تعریف توپولوژی شبکه، اجرای سوئیچ‌های برنامه‌پذیر و کنترل ترافیک داده فراهم شود. سوییچ‌ها و منطق پردازش بسته‌ها با استفاده از زبان P4 طراحی شدند و بخش یادگیری فدرال با زبان Python پیاده‌سازی گردید تا ارتباط میان گره‌های محاسباتی و سوییچ‌ها به صورت پویا برقرار شود. کلیه آزمایش‌ها در یک ماشین مجازی از پیش پیکربندی‌شده انجام شد که از سیستم‌عامل لینوکسی و قابلیت اتصال گرافیکی از طریق X-Server پشتیبانی می‌کرد. این محیط امکان ارزیابی دقیق عملکرد پروتکل طراحی‌شده در شرایط کنترل‌شده شبکه‌ای را فراهم نمود.

۳ مراحل انجام پروژه

۱.۳ راه اندازی زیرساخت پروژه

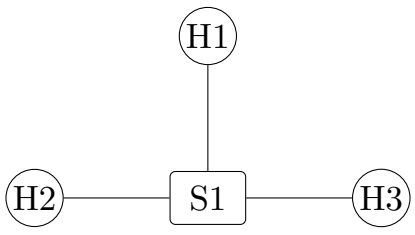
برای راه‌اندازی پروژه باید از ابزار P4-Utils استفاده کنید. می‌توانید این ابزار را بر روی دستگاه خودتان نصب کنید و یا از ماشین مجازی که به صورت از پیش آماده شده در سایت این ابزار موجود است استفاده کنید. این ماشین مجازی به صورت پیش‌فرض برای استفاده از Qemu آماده شده اما می‌توانید با استفاده از دستوراتی که در ضمیمه آ آمده فایل دیسک را به فرمت‌های قابل استفاده در دیگر مجازی‌سازها تبدیل کنید.

۲.۳ پیاده‌سازی توپولوژی

توپولوژی شبکه‌ای که هدف پروژه است در اختیار قرار شما گرفته. بر اساس ساختار مشخص‌شده، توپولوژی را در فایل `p4app.json` پیکربندی نموده و سپس با اجرای دستور

```
sudo p4run
```

در پوشه محتوی فایل `p4app.json` توپولوژی را در محیط مینی‌نت راه‌اندازی کنید.



۳.۳ پیاده‌سازی مسیریابی پایه در P4

سوییچ‌ها باید در گام نخست قادر به مسیردهی بسته‌های IP باشند. با تعریف هدر پروتکل‌های Ethernet،IPv4 پارسر و دیپارسر، منطق پردازش بسته در اینگرس و جداول مورد نیاز، منطق کنترلی را در `p4/federated_learning.p4` و مقداردهی جداول مور دنیاز برای مسیردهی در لایه شبکه در فایل `switch-commands.txt` مسیریابی در برنامه P4 پیاده‌سازی کنید. پس از انجام این کار از طریق اجرای مینی‌نت و اجرای دستور

```
pingall
```

از ارتباط بین بین پردازشگرها و عملکرد صحیح این مکانیزم اطمینان حاصل نمایید.

۴.۳ اندازه‌گیری عملکرد یادگیری فدرال در حالت مرسوم

فرایند یادگیری فدرال را به‌صورت استاندارد (یعنی حالتی که هر پردازشگر به‌طور مستقیم وزن‌های خود را برای سایر واحدها ارسال می‌کند) پیاده‌سازی نمایید (قسمت‌های مختلفی در فایل‌های درون فولدر `src` که با TODO مشخص شده‌اند را پیاده‌سازی نمایید.) سپس مینی‌نت را اجرا نموده و با دستور

```
xterm h{Node ID}
```

ترمینال هر پردازشگر را باز کرده و کد یادگیری ماشین را از طریق اجرای فایل `src/worker/worker.py` اجرا نمایید. (برای اتصال به پردازشگرها باید دسترسی به X server برقرار باشد. برای راهنمایی درباره چگونگی ایجاد این دسترسی به ضمیمه مراجعه ب نمایید.) سپس تعداد بسته‌های منتشر شده در شبکه را از طریق فایل‌های pcap ذخیره شده جمع‌آوری نموده و نمودار تعداد پیام‌ها را بر حسب تعداد دفعات تکرار یادگیری رسم کنید.

۵.۳ پیاده‌سازی پروتکل جمع‌آوری وزن‌ها

پروتوکل و ساز و کاری طراحی نمایید که در آن سوییچ پس از دریافت بسته‌های وزن ارسالی از جانب پردازشگرها، تا زمان دریافت تمامی بسته‌های موردنیاز وزن‌ها را در رجیسترهای سوییچ ذخیره کرده و منتظر بماند؛ سپس و با رسیدن تمامی بسته‌ها، مقادیر وزن ذخیره شده را تجمیع کرده و نتیجه جمع‌شده را برای تمامی پردازشگرها توزیع نماید. (این پروتکل را در کد P4 و Python پیاده‌سازی کنید.)
راهنمایی:

- برای ارسال وزن‌ها می‌توانید وزن‌ها را بین چند بسته شکسته و به جای ارسال یک بسته بزرگ چندین بسته کوچک ارسال کنید.

- برای اطمینان از اینکه همه پردازشگرها بسته خود را ارسال کرده‌اند می‌توانید به هر پردازشگر یک آیدی اختصاص دهید و با جمع کردن آیدی‌های آن‌ها در یک رجیستر در سوییچ با رسیدن به عدد هدف بسته تجمیع را ارسال نمایید.

- زبان P4 قابلیت کار با اعداد Floating point را ندارد و به همین خاطر نمی‌توان وزن‌ها را به صورت عادی برای سوییچ ارسال کرد. شبکه عصبی درون کد طوری تنظیم شده که وزن‌هایی بین $[-0.5, 0.5]$ داشته باشد. به همین خاطر می‌توانید با ضرب کردن مقدار وزن‌ها در یک عدد بزرگ و قطع کردن باقیمانده اعشاری آن‌ها را به عدد صحیح تبدیل کرده و سپس برای سوییچ ارسال کنید.

۶.۳ اندازه‌گیری حالت بهبود یافته با بهره‌گیری از پردازش درون‌شبکه

فرایند یادگیری فدرال را بار دیگر اجرا کنید، اما این‌بار با پروتکلی که در آن پردازشگرها وزن‌های محلی را به سوییچ میانی ارسال می‌کنند تا پردازش تجمیع در شبکه انجام شود. معیارهایی که در گام سوم جمع‌آوری شده بودند را مجدداً اندازه‌گیری کرده و نتایج را با حالت اولیه مقایسه کنید؛ سپس بهبود مشاهده‌شده را تحلیل نموده و درباره آن بحث نمایید.

۴ تحویل دادنی‌ها

درنهایت یک فایل PDF حاوی تصویر مراحل مختلف انجام پروژه را به همراه کدهایی که برای یادگیری فدرال عادی و کدهای یادگیری فدرال با استفاده از سوییچ پیاده کردید را در قالب یک فایل با فرمت Zip تحویل دهید.

آ دستورات تبدیل دیسک ماشین مجازی

برای تبدیل دیسک ماشین مجازی از فرمت qcow2 به فرمت‌های مورد پشتیبانی VMware و VirtualBox می‌توانید از دستورات زیر استفاده کنید:

Listing 1: To VMDK (For VMware)

```
qemu-img convert -f qcow2 p4-vm.qcow2 -O vmdk p4-vm.vmdk
```

Listing 2: To VDI (For VirtualBox)

```
qemu-img convert -f qcow2 p4-vm.qcow2 -O vdi p4-vm.vdi
```

ب فعال‌سازی X server برای اتصال به پردازشگرها

برای فعال‌سازی اتصال به X server پس از اتصال ماشین مجازی به شبکه، با اضافه کردن فلگ `-X` به دستور اتصال SSH از طریق لینوکس و یا WSL به ماشین مجازی متصل شوید. سپس دستور

Listing 3: To VMDK (For VMware)

```
sudo xauth merge /home/p4/.Xauthority
```

دسترسی X server را برای کاربر روت فعال کرده و سپس به ادامه انجام پروژه بپردازید.

در صورت بروز هرگونه مشکل، پرسش یا ابهام، می‌توانید از طریق این آدرس ایمیل با دستیار آموزشی مسئول این پروژه در ارتباط باشید و موضوع را مطرح نمایید.

موفق باشید