طراحی کامپیوتری سیستم‌های دیجیتال

دانشکده مهندسی برق و کامپیوتر

پاییز ۱۴۰۲

پروژه اول: شبکه عصبی Maxnet

دستیاران آموزشی: آوا میرمحمدمهدی، سارا رضائی‌منش، نسا عباسی

توضیحات پروژه

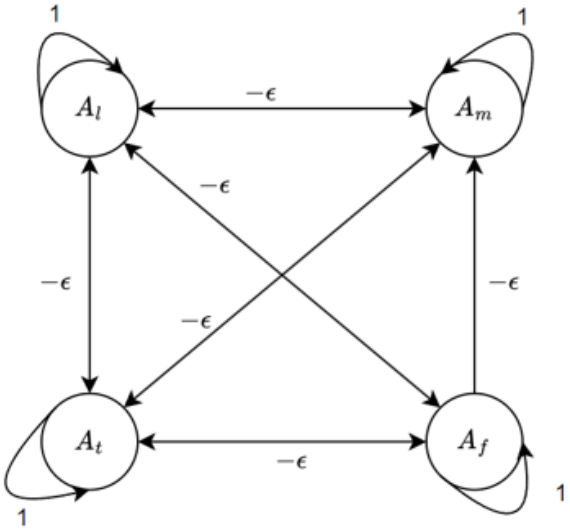
شبکه عصبی Maxnet در دسته‌بندی شبکه‌های یادگیری رقابتی قرار می‌گیرد. این شبکه در دهه ۱۹۸۰ توسط جیمز کارپنتر و استفن گروسبرگ پیشنهاد و به عنوان یک شبکه عصبی رقابتی معرفی شد.

شبکه‌های عصبی رقابتی عموما برای دسته‌بندی و کاهش بعد مورد استفاده قرار می‌گیرند. این شبکه ها با افزودن یک ساختار اضافه به شبکه‌های عصبی معمولی، شرایطی را ایجاد می‌کنند که در آن تنها یک نورون از چندین نورون موجود در یک گروه، سیگنال خروجی غیرصفر داشته باشد یا به عبارتی فعال شود. پس از گذراندن نقاط داده از شبکه عصبی رقابتی، نقاطی که نورون یکسانی را فعال می‌کنند، در یک دسته قرار می‌گیرند.

Maxnet ، نمونه‌ای خاص از یک شبکه عصبی بر پایه رقابت است. یکی از کاربردهای این شبکه، یافتن Max از یک مجموعه مقادیر عددی‌ست. فرض کنیم در این شبکه با چهار نورون می خواهیم بزرگترین مقدار ورودی های x_1, x_2, x_3, x_4 را پیدا کنیم. در این مدل برخالف شبکه عصبی که در درس هوش مصنوعی یادگرفتید، نورون ها از هم مستقل نبوده و با هم در رقابت هستند. این شبکه را به صورت زیر تعریف می کنیم:

$$w_{ij} = \begin{cases} 1, & i = j \\ -\epsilon, & otherwise \end{cases}$$

$$\text{Activation function: } f(x) = ReLU(x) = \begin{cases} x, & x > 0 \\ 0, & otherwise \end{cases}$$



شکل ۱. معماری شبکه Maxnet

در هر دوره الگوریتم زیر بر روی شبکه داده شده اجرا می‌شود:

۱. activation ها و weight ها را به صورت زیر مقداردهی کنید:
   a. x_j را مقدار گره A_j قرار دهید.
   b. وزن شبکه را به صورت زیر مقداردهی کنید:

$$w_{ij} = \begin{cases} 1, & i = j \\ -\epsilon, & otherwise \end{cases}$$

۲. مقدار activation هر گره $j = 1, \ldots M$ را به صورت زیر آپدیت کنید:

$$a_j^{new} = f\left[a_j^{old} - \epsilon \sum_{k \neq j} a_k^{old}\right]$$

۳. مقدار activation های جدید را برای استفاده در دوره بعد ذخیره کنید.
۴. تا وقتی بیش از یک گره با مقدار غیرصفر وجود دارد به گام ۲ برو و activation ها را آپدیت کن. در صورتی که فقط یک گره با مقدار غیرصفر ماند، محاسبات را متوقف کن.

گره‌ای که مقدرا عددی غیرصفر دارد، به عنوان گره برنده انتخاب می‌شود. این گره نشان‌دهنده بزرگترین عدد بین x_1 تا x_4 است.

برای درک این شبکه به مثال زیر توجه کنید:

یک شبکه Maxnet با چهار نورون و پارامتر $\epsilon = 0.2$ و مقادیر نورون‌های زیر در نظر بگیرید.

$$x_1 = 0.2 \qquad x_2 = 0.4 \qquad x_3 = 0.6 \qquad x_4 = 0.8$$

یک دوره از محاسبات شبکه به صورت زیر است:

$$a_j^{new} = f\left[a_j^{old} - \epsilon \sum_{k \neq j} a_k^{old}\right]$$

$$a_1^{new} = f\left[a_1^{old} - 0.2\left(a_2^{old} + a_3^{old} + a_4^{old}\right)\right]$$
$$a_2^{new} = f\left[a_2^{old} - 0.2\left(a_1^{old} + a_3^{old} + a_4^{old}\right)\right]$$
$$a_3^{new} = f\left[a_3^{old} - 0.2\left(a_1^{old} + a_2^{old} + a_4^{old}\right)\right]$$
$$a_4^{new} = f\left[a_4^{old} - 0.2\left(a_1^{old} + a_2^{old} + a_3^{old}\right)\right]$$

$$a_1^1 = f[a_1^0 - 0.2(a_2^0 + a_3^0 + a_4^0)] = f[0.2 - 0.2(0.4 + 0.6 + 0.8)] = f[-0.16] = 0$$
$$a_2^1 = f[a_2^0 - 0.2(a_1^0 + a_3^0 + a_4^0)] = f[0.2 - 0.2(0.2 + 0.6 + 0.8)] = f[0.08] = 0.08$$
$$a_3^1 = f[a_3^0 - 0.2(a_1^0 + a_2^0 + a_4^0)] = f[0.2 - 0.2(0.2 + 0.4 + 0.8)] = f[0.32] = 0.32$$
$$a_4^1 = f[a_4^0 - 0.2(a_2^0 + a_3^0 + a_4^0)] = f[0.2 - 0.2(0.2 + 0.4 + 0.6)] = f[0.56] = 0.56$$

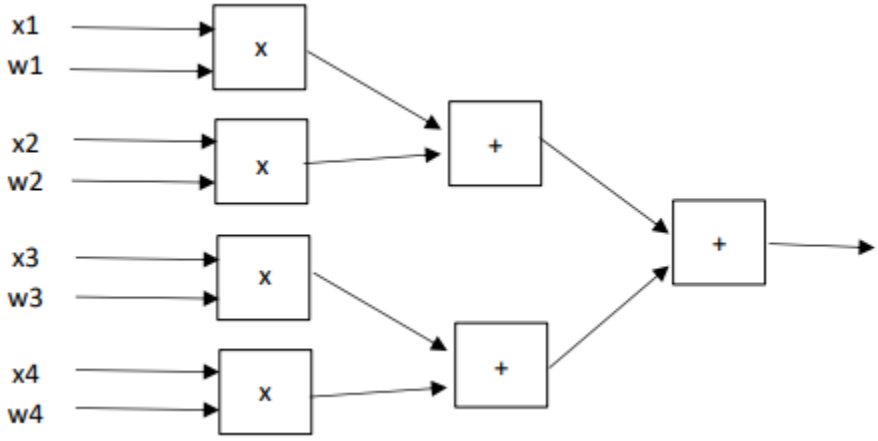
در دوره آخر، مقدار همه نورون ها به جز نورون چهارم صفر می‌شود که یعنی x_4 بزرگترین مقدار بوده.

$$a_1^1 = 0.0 \qquad a_2^1 = 0.08 \qquad a_3^1 = 0.32 \qquad a_4^1 = 0.56$$
$$a_1^2 = 0.0 \qquad a_2^2 = 0.0 \qquad a_3^2 = 0.192 \qquad a_4^2 = 0.48$$
$$a_1^3 = 0.0 \qquad a_2^3 = 0.0 \qquad a_3^3 = 0.096 \qquad a_4^3 = 0.442$$
$$a_1^4 = 0.0 \qquad a_2^4 = 0.0 \qquad a_3^4 = 0.008 \qquad a_4^4 = 0.422$$
$$a_1^5 = 0.0 \qquad a_2^5 = 0.0 \qquad a_3^5 = 0.0 \qquad a_4^5 = 0.421$$

جزئیات پیاده‌سازی

شبکه Maxnet ذکر شده به صورت مجموعه‌ای از واحدهای پردازش (PU) و یک کنترل‌کننده پیاده‌سازی می‌شود. هر واحد PU یک واحد ضرب و جمع است که هشت ورودی دارد. این واحد شامل چهار ضرب‌کننده، یک درخت جمع‌کننده و یک تابع فعال‌سازی است.

- عملیات PU شامل چهار مرحله است که هر کدام به اندازه یک کلاک سایکل طول می‌کشند. این چهار مرحله عبارتند از: خواندن داده از حافظه، ضرب‌کننده، درخت جمع‌کننده + فعال‌سازی و نوشتن نتایج در رجیسترها.
- جهت صرفه‌جویی در مصرف منابع، فرض کنید تنها چهار PU در اختیار دارید و محاسبات هر دوره به صورت سریالی انجام می‌شود.
- برای محاسبات اعشاری و ذخیره اعداد اعشاری، از استاندارد IEEE ۷۵۴ استفاده کنید.
- برای واحدهای ضرب کننده، می‌توانید عمل ضرب را در سطح بالا (a*b) پیاده‌سازی کنید. (توجه داشته باشید که برای ضرب دو عدد هشت بیتی با نمایش علامت-مقدار، باید از ضرب بی‌نشان 7*7 استفاده کنید و مقدار بیت علامت را به صورت جداگانه کنترل کنید.)
- برای ذخیره وزن های شبکه از بافرهای حافظه استفاده کنید.
- در این پروژه مقدار $\epsilon = 0.2$ در نظر بگیرید.
- یک سیگنال خروجی done در نظر بگیرید که نشان دهد محاسبات اتمام یافته است.
- تمام بافرها را ۳۲ بیتی در نظر بگیرید.



شکل ۲. ساختار PU

نکات پایانی

- انجام این تمرین به صورت گروه‌های دونفره خواهد بود.
- این پروژه باید در دو فاز انجام شود:
  ۱. در فاز اول controller و datapath را طراحی کرده و در موعد تعیین شده برای فاز اول داخل سایت آپلود کنید.
  ۲. در فاز دوم datapath و controller طراحی شده در فاز اول را در مادل سیم و با زبان وریلاگ پیاده‌سازی کرده و در موعد تعیین شده برای فاز دوم داخل سایت آپلود کنید.
- برای فاز دوم تمرین، لازم است فایل های HDL و testbench خود را مطابق ساختار توضیح داده شده در trunk/doc در subdirectory های trunk آپلود کنید. همچنین، اطمینان حاصل کنید که با اجرای trunk/sim/sim_top.tcl تست بنچ شما اجرا می‌شود. برای اجرای این اسکریپت میتوانید از دستور زیر در Modelsim استفاده کنید:

```
>> do <sim_file>
```

- فایل ها و گزارش خود را تا قبل از موعد تحویل هر فاز، با نام‌های CAD_HW1_P1_<SID>.zip و CAD_HW1_P2_<SID>.zip به ترتیب در محل های مربوطه برای فاز اول و دوم در صفحه درس آپلود کنید.
- هدف از این تمرین، یادگیری شماست! در صورت کشف تقلب، مطابق با قوانین درس برخورد خواهد شد.