KNO-1001-3904

# تعیین بهترین شبکه MLP برای تخمین توابع ریاضی

امین شیخ نجدی
aminnima2@gmail.com

محمدحسین رضوانی
Mhrezvani2003@yahoo.com

**چکیده:** ما در این مقاله سعی داریم روشی برای پیداکردن بهترین شبکه MLP که می‌تواند یک تابع را تخمین بزند ارائه دهیم در این روش با استفاده نرم‌افزار مطلب ابتدا یک شبکه MLP را پیاده‌سازی می‌کنیم سپس عملکرد آن را به‌ازای تعداد نرون‌های مختلف همچنین به‌ازای تعداد لایه‌های مختلف بررسی می‌کنیم، در هر مرحله تفاضل خروجی شبکه را از خروجی مطلوب به‌عنوان خطای شبکه در نظر می‌گیریم و در پایان عملکرد یک شبکه مناسب را به‌عنوان معیاری برای تعیین مطلوب‌بودن شبکه ارائه می‌دهیم.

**کلید واژه:** *داده تست(test) , داده آموزش(train) ,تابع انتقال, نرون.*

## ۱- مقدمه

امروزه کاربردهای شبکه های عصبی در زمینه های مختلف گسترش‌یافته اما موضوعی که بسیار مهم می‌باشد پیداکردن شبکه منطبق بر توپولوژی مسئله است.

طراحی یک شبکه بر پایه نوع داده‌های مسئله، گستردگی ورودی‌ها و رنج تغییرات ورودی انجام می‌شود، اما گاهی گستردگی مسئله به‌گونه‌ای است که توپولوژی مسئله از روی داده‌های ورودی قابل فهمیدن نیست و یا اینکه تمام داده‌ها را در اختیار نداریم.

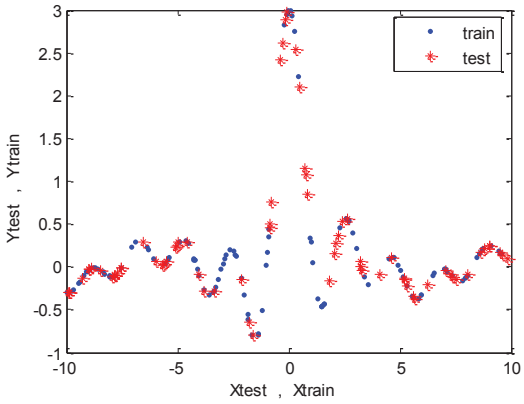
رو شی که در این مقاله ارائه شده به‌گونه‌ای ا ست که کاربر بتواند با داشتن بخش کوچکی از داده‌های مسئله مناسب‌ترین شبکه را پیدا کند.

همواره برای دستیابی به یک شبکه مناسب برای شروع یک شبکه ابتدایی تعریف می‌کنند و سپس با استفاده از داده‌های مسئله شبکه را بهینه می‌کنیم، ما نیز در این مقاله ابتدا یک شبکه ابتدایی تعریف کرده و سپس با استفاده از داده‌های مسئله، تعداد توپولوژی مناسب شبکه را پیدا می‌کنیم و در نهایت خروجی بهترین شبکه به‌دست‌آمده را ارائه می‌دهیم.

## ۲- معرفی تابع

در ابتدا ما دو نوع داده data test,data train را برای بررسی عملکرد شبکه در نظر می‌گیریم. از داده‌ها (train) برای آموزش شبکه استفاده می‌کنیم و از داده‌های (test) به‌عنوان معیاری برای تعیین کیفیت عملکرد مطلوب شبکه بهره می‌گیریم. اگر داده‌های test,train را در نرم‌افزار مطلب رسم کنیم شکل تابع موردنظر به‌صورت شکل (۱) خواهد شد

همان‌طور که در شکل (۱) مشاهده می‌کنید نقاط آبی نمایانگر داده‌های (train) و ستاره‌ها نشان‌دهنده داده‌های (test) در شکل فوق داده‌های مسئله بین ( 10تا10-) تغیر می‌کند.



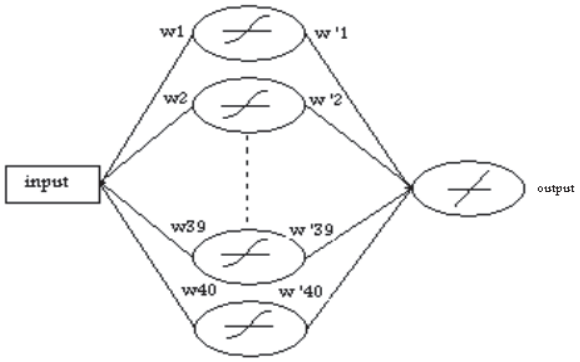

**شکل ۱:** داده‌های train ,test

## ۳- معرفی شبکه

در این قسمت می‌خواهیم یک شبکه MLP را با استفاده از دستور newff در مطلب پیاده‌سازی کنیم

Net=newff([-10 10],[40 1],{‘tansig’,’purelin’})

شبکه فوق یک شبکه MLP با دولایه است که در لایه اول 40 نرون وجود دارد تابع انتقال پنج نرون اول از نوع (tansig) است و تابع انتقال لایه خروجی از نوع (purelin) است.شکل (۲)تابع فوق را نشان می‌دهد.




**شکل۲: شبکه MLP**

## ۴- طراحی مناسب شبکه

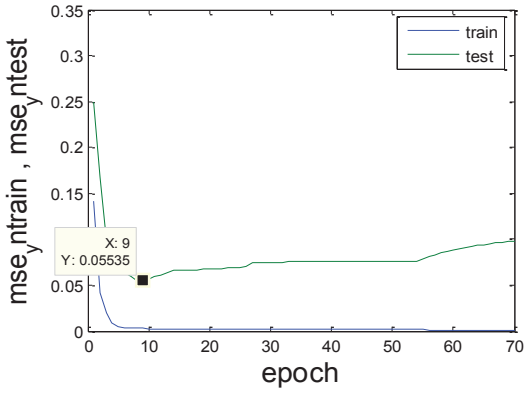
برای طراحی یک شبکه مناسب باید تعداد نرون‌ها و تعداد لایه‌ها بهینه شود همچنین باید شبکه به طور مطلوب آموزش ببینند.

ابتدا برنامه‌ای در مطلب می‌نویسیم که تعداد نرون‌های لایه اول را برای ما بهینه کند تکه برنامه زیر شبکه را با استفاده از دستور net.trainparam.epches به‌ازای مقادیر متفاوت epoch(هر دور آموزش به‌ازای داده‌های train ) آموزش می‌دهد و سپس در هرepoch خطای داده‌های train, test را محاسبه کرده در نهایت نمودار شکل (۳) به دست می‌آید.

```
for z=1:1:70
net=newff([-10 10],[40 1],{'tansig','purelin'});
net.trainparam.epochs=z;
net=train(net,xtrain',ytrain');
yntrain=sim(net,xtrain');
yntest=sim(net,xtest');
mse_yntraina(z)=sqrt((yntrain-ytrain')*(yntrain-
ytrain')'/length(ytrain));
mse_yntesta(z)=sqrt((yntest-ytest')*(yntest-
ytest')'/length(ytest));
end
figure(2);
plot([mse_yntraina',mse_yntesta']);
legend('train','test');
```

همان‌طور که در شکل (۳) مشاهده می‌شود با افزایش تعداد epoch خطای داده‌های train کم شده؛ اما خطای داده‌های test از حد معینی شروع به افزایش کرده در این حالت شبکه دچار over train شده.

مطابق شکل بهترین تعداد Epoch برای این شبکه برابر (۹)است.




**شکل۳: نمودار بهینه‌کردن تعداد epochها**

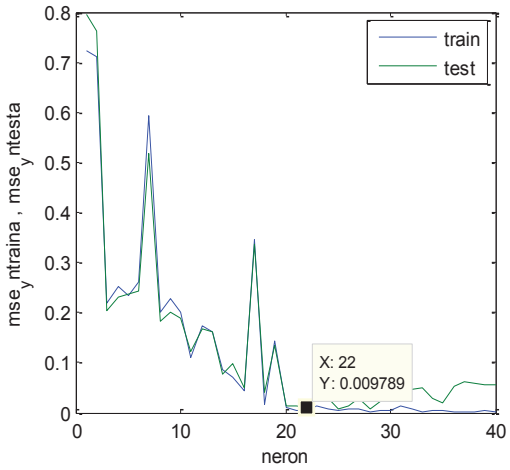
علت این که تعداد epoch های بهینه را(۹) انتخاب کردیم این است که در این حالت خطای داده‌های test کمترین مقدار را دارد و خطای داده‌های train نیز با افزایش تعداد epochها تغیر چندانی ندارد.

در مرحله بعدی می‌خواهیم تعداد نرون‌های لایه اول را به همان ترتیبی که تعداد epochها را بهینه کردیم به دست آوریم درواقع در حالت قبلی ما متغیر برنامه را تعداد epoch در نظر گرفته بودیم اما در این مورد متغیر را تعداد نرون لایه دوم در نظر می‌گیریم برنامه زیر این کار را انجام می‌دهد.

```
for z=1:1:70
net=newff([-10 10],[z 1],{'tansig','purelin'});
net.trainparam.epochs=50;
net=train(net,xtrain',ytrain');
```

```
yntrain=sim(net,xtrain');
yntest=sim(net,xtest');
mse_yntraina(z)=sqrt((yntrain-ytrain')*(yntrain-
ytrain')'/length(ytrain));
mse_yntesta(z)=sqrt((yntest-ytest')*(yntest-
ytest')'/length(ytest));
end
figure(2);
plot([mse_yntraina',mse_yntesta']);
legend('train','test');
```

با اجرای برنامه فوق نمودار شکل (۴) به دست می‌آید در این حالت همان‌طور که از روی مشخص است تعداد نرون بهینه برای لایه اول برابر ۲۲ است.



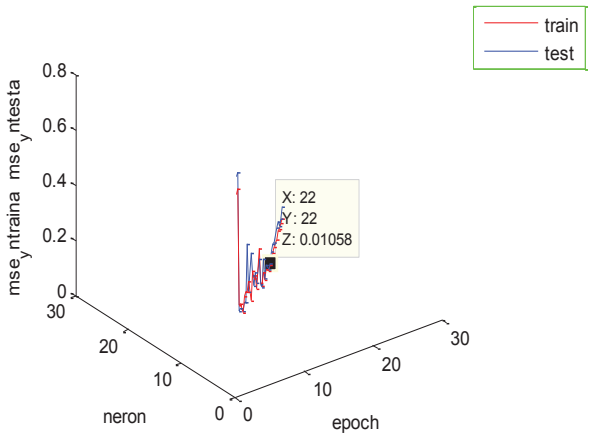

**شکل ۴: نمودار بهینه‌کردن تعداد نرون‌ها**

اشکال دو برنامه فوق این است که ما تعداد epoch=9 را در حالتی به دست آورده‌ایم که تعداد نرون‌ها برابر ۴۰ باشد و همچنین در حالت دوم تعداد نرون بهینه در وضعیتی به‌دست‌آمده که تعداد epoch برابر ۵۰ باشد برای رفع مشکل فوق‌برنامه بالا را به صورتی تغییر می‌دهیم که تعداد نرون‌های لایه اول همزمان با تعداد epoch بهینه شود این کار به کمک دو حلقه for تودرتو به‌صورت زیر میسر است.

```
for z=1:1:30
for k=1:1:30
net=newff([-10,10],[z,1],{'tansig','purelin'});
net.trainparam.epochs=k;
net=train(net,xtrain',ytrain');
yntrain=sim(net,xtrain');
yntest=sim(net,xtest');
mse_yntraina(z)=sqrt((yntrain-ytrain')*(yntrain-
ytrain')'/length(ytrain));
mse_yntesta(z)=sqrt((yntest-ytest')*(yntest-
ytest')'/length(ytest));
end
end
```

```
figure(3);
z=1:1:30;
k=1:1:30;
plot3(k,z,mse_yntraina);
hold on
plot3(k,z,mse_yntesta);
legend('train','test');
```

حال اگر برنامه فوق را اجرا کنیم نمودار سه‌بعدی شکل(5) حاصل می‌شود.




شکل ۵:نمودار بهینه‌کردن تعداد نرون‌هایepochها

در این شکل دو نمودار متفاوت وجود دارد که یک خطای داده‌های تست را حساب می‌کند و دیگری خطای داده‌های train را محاسبه می‌کند از دو حالت قبلی به یاد داریم که با افزایش تعداد نرون و epoch خطای داده‌های train کاهش می‌یابد پس اگر ما خطای داده‌های test را بررسی کنیم خواهیم دید که عدد ۲۲ مناسب‌ترین مقدار برای نرون‌ها و epochها است.

حال که تعداد نرون بهینه را هم‌زمان با تعداد epoch بهینه به دست آورده‌ایم , می‌خواهیم وضعیت توابع انتقال را بررسی کنیم، انتخاب یک تابع انتقال مناسب علاوه بر اینکه به نوع مسئله بستگی دارد به لایه‌ای که در آن استفاده می‌شود نیز بستگی دارد؛ مثلاً اگر در شبکه فوق بخواهیم تابع انتقال لایه اول را تغیر دهیم و بجای آن از logsig استفاده کنیم نتایجی متفاوت باحالت قبل به دست خواهیم آورد همچنین اگر از تابع sign استفاده کنیم مقادیر به‌دست‌آمده متفاوت خواهد بود.

روشی که برای انتخاب یک تابع انتقال مناسب پیشنهاد می‌کنیم این است که مثلاً خطای نرون بهینه را برای دو تابع متفاوت باهم بررسی کنیم در حالت اول تعداد نرون بهینه‌ای که به دست آوردیم ۲۲ با خطای معدل 0.0097 بود حال شبکه را با چند تابع انتقال مختلف بررسی می‌کنیم.

Logsig

mse_yntesta=0.0077

neron=23 .

satlins

mse_yntesta=0.0349

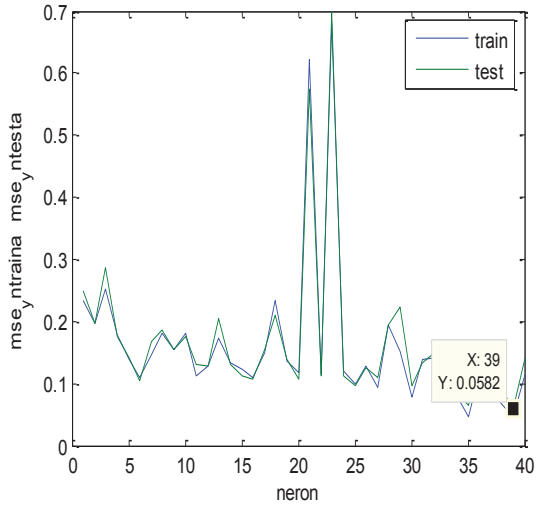
neron=23

همان‌طور که از مقادیر بالا مشخص است توابع انتقال متفاوت تقریباً تعداد نرون یکسانی به دست می‌آورند؛ اما خطای آنها متفاوت است در واقع مناسب‌ترین تابع انتقال برای مسئله فوق logsig است با انجام عملیات فوق برای لایه دوم بهترین تابع انتقال purelin به دست می‌آید

در ادامه می‌خواهیم وضعیت تعداد لایه‌ها را بررسی کنیم برای این کار شبکه زیر را معرفی می‌کنیم

Net=newff([-10 10],[40 z 1],{tansig 'tansig','purelin'})

دارای یک شبکه مخفی است که تعداد آن را به‌صورت متغیر z در نظر می‌گیریم حال اگر بخواهیم شبکه را مطابق برنامه بالا اجرا کنیم تعداد نرون لایه مخفی از روی شکل (۶) برابر ۳۹ به دست می‌آید




شکل ۶:نمودار تعداد نرون لایه مخفی

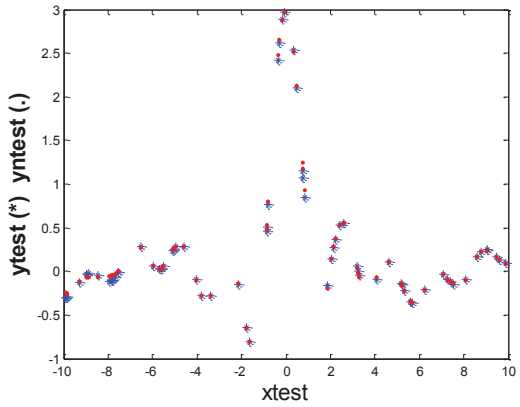
به‌وضوح مشخص است که افزایش یک‌لایه به‌هیچ‌وجه برای شبکه مناسب نیست و فقط با افزایش خطا همراه خواهد بود، البته این ویژگی مختص داده‌های این مسئله است و ممکن است به‌ازای داده‌های یک مسئله دیگر افزایش یک‌لایه مفید باشد.

حال که شبکه را از جهات مختلف بررسی کرده‌ایم می‌خواهیم شبکه را با اطلاعاتی که به دست آورده‌ایم اجرا کنیم

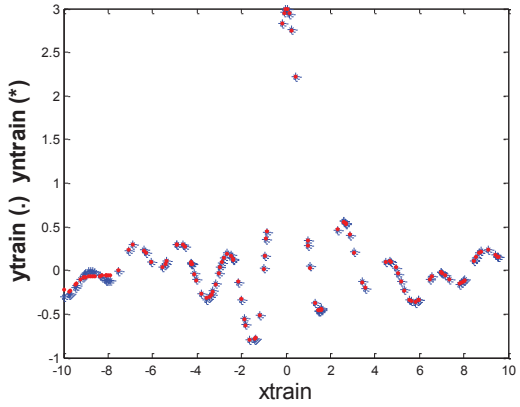
اجرای شبکه با استفاده از دستور زیر انجام می‌شود

```
yntrain=sim(net,xtrain');
yntest=sim(net,xtest');
```

درواقع یک‌بار شبکه را با داده‌های test و یک‌بار با داده‌های train اجرا می‌کنیم و سپس خروجی آن را روی خروجی مطلوب رسم می‌کنیم انتظار داریم که هر دودسته نقاط روی‌هم منطبق شوند. شکل‌های (۷ و ۸) به ترتیب خروجی شبکه را بر روی داده‌های test و train منطبق نموده‌اند




شکل ۷:نمودار خروجی شبکه برای داده‌های test




شکل ۸:نمودار خروجی شبکه برای داده‌های train

حالت فوق نشان‌دهنده این موضوع است که شبکه به طور مناسب طراحی شده وضعیت فوق را می‌توان به‌عنوان معیاری برای تعین عملکرد مطلوب شبکه در نظر گرفت.

**نتیجه‌گیری**

بررسی‌های فوق نشان می‌دهد که بهینه‌کردن شبکه به عوامل متفاوتی بستگی دارد و تعیین مناسب‌ترین پارامترها برای شبکه گاهی نیاز به تحلیل و بررسی هر کدام به‌صورت جداگانه دارد و همچنین تأثیر هر کدام بر روی خروجی شبکه متفاوت است در نهایت این دقت موردنیاز کاربر است که مشخص می‌کند شبکه باید تا چه حد بهینه شود؛ اما نکته مهم این است که حتی کسانی هم که به صورت حرفه‌ای با شبکه‌ها آشنایی ندارند نیز می‌توانند با استفاده از این روش شبکه مناسب برای کار خود را به دست آورند.

## مراجع

[1] Randy L. Haupt and Sue Ellen, Haupt ,"PRACTICALGENETIC ALGORITHMS " , JOHN WILEY & SONS, INC., PUBLICATION,2004

[2] Oliver nelles,"Nonlinear System identification" , springer-verlag berlin heidelberg New york 2001

.