به نام آنکه آموخت انسان را آنچه نمی دانست

دانشگاه تهران

پردیس دانشکده‌های فنی

دانشکده برق و کامپیوتر

درس پردازش زبان‌های طبیعی

CA3

Feed Forward Neural Network

behzad.shayegh@ut.ac.ur

بهزاد شایق بروجنی

810 196 678

فروردین ۱۳۹۹

فهرست

فهرست 1

مقدمه 2

دادگان 2

پیش‌پردازش 2

بخش اول) استفاده از لایه‌ی پیش‌آموزش داده شده. 2

شرح کلی پیاده‌سازی 2

نمودار مقادیر Perplexity 3

بررسی تأثیر اندازه‌ی پنجره 3

بررسی تأثیر گام آموزش 5

بررسی تأثیر نورون‌های مخفی 7

بخش دوم) آموزش بردار کلمات در شبکه. 9

شرح کلی پیاده‌سازی 9

نمودار مقادیر Perplexity 10

فایل‌های جانبی 11

مقدمه

با سلام. در این تمرین قصد داریم با استفاده از شبکه‌ی عصبی feed forward یک مدل زبانی پیاده‌سازی کنیم و تأثیر پارامترهای متفاوت آن را بررسی کنیم.

دادگان

دادگان مورد استفاده در این آزمایش یک متن بزرگ برای Train و یک متن کوچک برای Test می‌باشد.

پیش‌پردازش

در مرحله‌ی پیش‌پردازش این آزمایش، ما مجموعه لغات GloVe.6B را معیار قرار داده و پیش‌پردازش را بر روی لغات خارج این مجموعه اعمال کردیم. این پیش‌پردازش شامل اعمالی همچون حذف کاراکترهای خاص، lemmatize , stem ، ریشه‌یابی دستی و ... بود. هر مرحله پردازش فقط بر روی لغاتی که در مجموعه GloVe.6B یافت نمی‌شدند انجام می‌گرفت. در پایان نیز که به هیچ‌وجه لغت مشابه در دادگان پیدا نشد، از ‹UKN› بجای آن لغت استفاده کردیم و در نتیجه هیچ لغتی خارج GloVe.6B باقی نماند. تعداد ‹UNK›ها نیز انگشت شمار بود و در نتیجه در کل فرایند آن‌ها را فقط به شکل ‹UNK› درنظر گرفتیم. در نهایت، تعداد لغات یکتای ما به 59504 عدد رسید که عددی بزرگ و نشان‌دهنده‌ی دقت در تشخیص لغات است.

بخش اول) استفاده از لایه‌ی پیش‌آموزش داده شده.

شرح کلی پیاده‌سازی

برای پیاده‌سازی شبکه عصبی، از کتابخانه‌ی pytorch استفاده شد. در این پیاده سازی ابتدا کل دادگان را به بخش‌هایی window+1 کلمه‌ای بخش بندی کردیم (این تقسیم بندی دارای overlap بود) که window کلمه اول برای دادگان ورودی و کلمه‌ی آخر به عنوان label درنظر گرفته شد. سپس به هر کلمه‌ی موجود در دادگان یک عدد یکتا نسبت داده شد و این عدد جایگزین کلمه‌ی label شد. کلمات window را نیز با بردار GloVe آنها جایگزین کرده و با یک عملیات reshape به یک بردار ورودی 50*window بعدی رسیدیم. سپس این بردار به عنوان ورودی به شبکه داخل شده و پس از عبور از دو لایه (یک لایه پنهان و یک لایه خروجی) به یک بردار v بعدی تبدیل می‌شود که v تعداد کلمات یکتای مجموعه لغات هستند. از توابع tanh و sigmoid به ترتیب به عنوان توابع فعالسازی لایه‌های شبکه استفاده شد. در نهایت نیز از cross-entropy به عنوان loss استفاده کردیم و $perplexity = 2^{loss}$ را محاسبه کردیم. برای هر epoch آموزش شبکه، مجموعه‌ی حاوی بخش‌های window+1

کلمه‌ای را shuffle کردیم تا ترتیب عبارات ورودی به شبکه تا حد ممکن بی‌تأثیر باشد. شرط توقف نیز طی تعداد ثابت ۳۰۰ epoch است. نحوه به‌روز رسانی پارامترها نیز روش backpropagation با استفاده از gradient descent است.

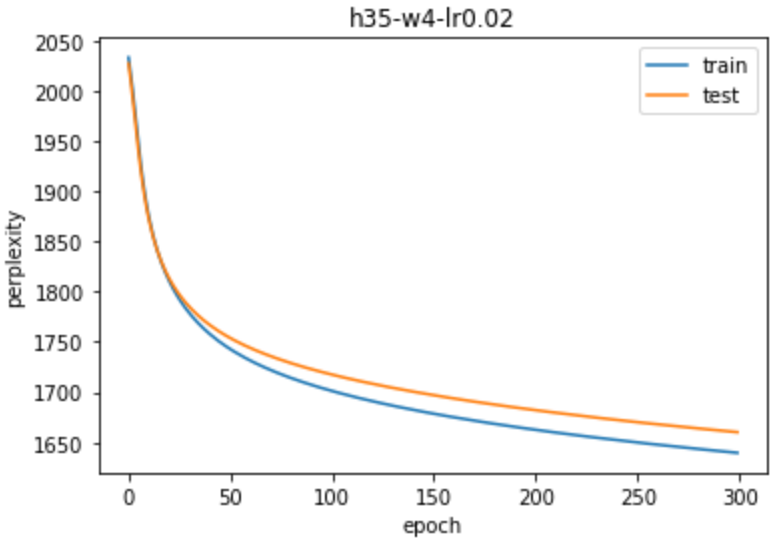
نمودار مقادیر Perplexity

با هایپر پارامترهای زیر، نمودار Perplexity برای iterationهای مختلف به شکل زیر در آمد:

Hidden layer neurons number = 35

Window size = 4

Learning rate = 0.02



همانطور که مشاهده می‌کنیم، این نمودار نشان می‌دهد که آموزش شبکه در ابتدا به سرعت انجام شده و به مرور شیب بهبود آن کاهش پیدا می‌کند و این به معنای نزدیک شدن به خطای کمینه‌است. همچنین نکته‌ی دیگری که قابل توجه است موضوع اختلافی است که نمودار دادگان تست به مرور از نمودار دادگان آموزش پیدا می‌کند و خطای آموزش بیشتر از خطای تست کاهش پیدا می‌کند و این به معنای حرکت به سمت overfit شدن است. به صورت کلی با یک نظر به نمودار متوجه می‌شویم که آموزش این شبکه در کل روند خوبی دارد اما متأسفانه خیلی پیشرفت نمی‌کند و Perplexity از حدود 42000 پایین‌تر نخواهد آمد.

اعداد بزرگ Perplexity به دلیل بزرگ بودن مجموعه لفات خروجی و درنتیجه بزرگ شدن ماتریس احتمالات خروجی و بزرگ شدن Cross entropy می‌باشد. پس اگر ما مجموعه کوچک‌تری را درنظر می‌گرفتیم (لغات کم تکرار را درنظر نمی‌گرفتیم یا در مرحله‌ی پیش‌پردازش ساده‌سازی بیشتری انجام می‌دادیم) Perplexity ما کاهش پیدا می‌کرد اما این به معنای موفقیت بیشتر نیست زیرا در این حالت که مجموعه لغات ما بزرگ‌تر است یعنی لغت خروجی را دقیق‌تر تعیین می‌کنیم.

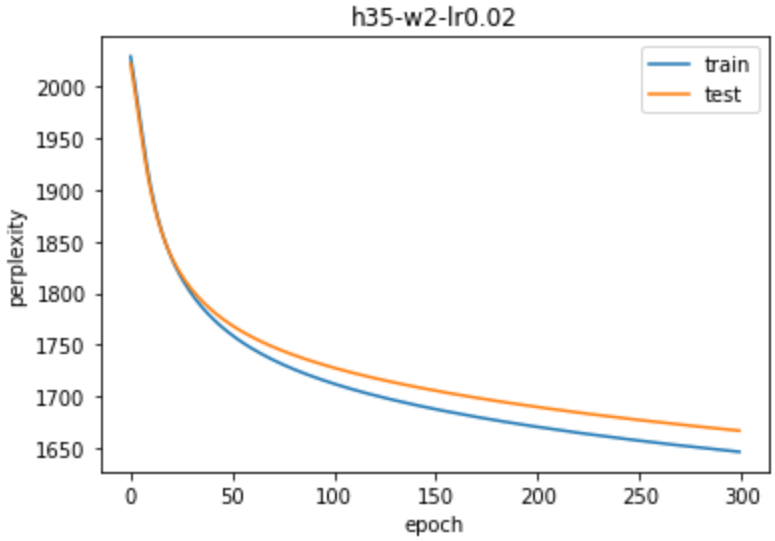
بررسی تأثیر اندازه‌ی پنجره

به دو نمودار زیر با هایپر پارامترهای مکتوب توجه کنید:

Hidden layer neurons number = 35

Window size = 2

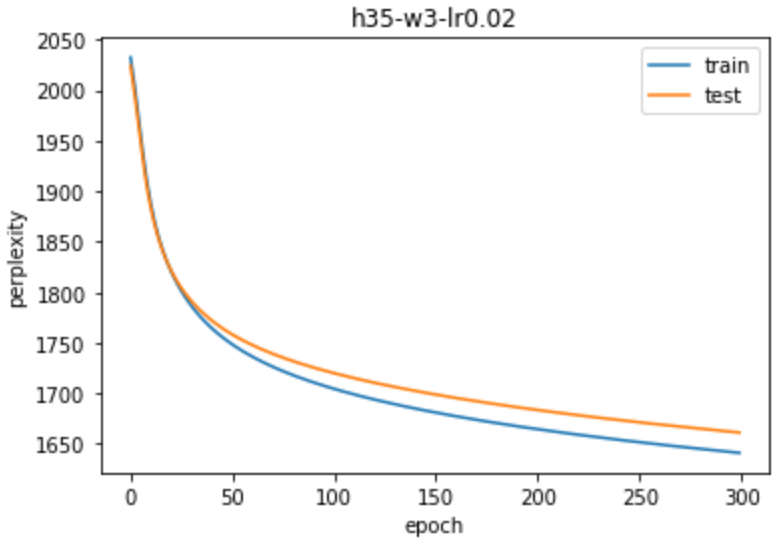
Learning rate = 0.02



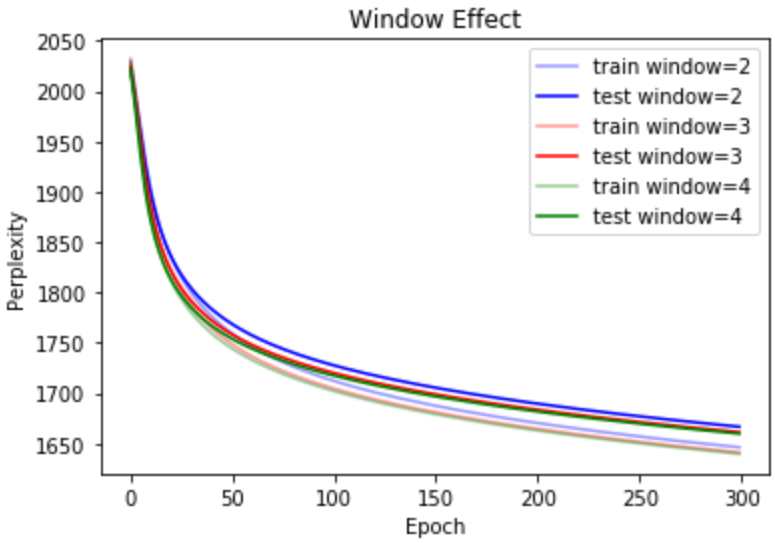
Hidden layer neurons number = 35

Window size = 3

Learning rate = 0.02



مقایسه‌ی نمودارهای فوق (به همراه نمودار بخش تحلیل Perplexity) را می‌توانید در نمودار زیر مشاهده کنید.



با مقایسه‌ی این دو نمودارها متوجه می‌شویم (با اختلاف کمی که تشخیص آن در نمودارهای بالا نیازمند دقت زیادیست) که نمودار perplexity با اندازه‌ی پنجره‌ی بزرگ‌تر با تقعر بیشتری نزول می‌کند (سریع‌تر آموزش داده می‌شود) اما اگر epochهای زیادی طی شود، همه‌ی مدل‌ها به دقت تقریبا برابری نزدیک می‌شوند (البته دقت پنجره‌ی بزرگ‌تر همواره بیشتر خواهد بود و با بررسی دقیق نمودارها این مورد مشخص است) و این به آن معناست که فاصله‌ی لغات تا لغت هدف با تاثیرشان بر احتمال حضور کلمات رابطه‌ی عکس دارد و عمده‌ی اطلاعات لازم برای پیشبینی یک لغت به لغات نزدیک به آن قابل یافت است و این اثر مارکوف را نشان می‌دهد.

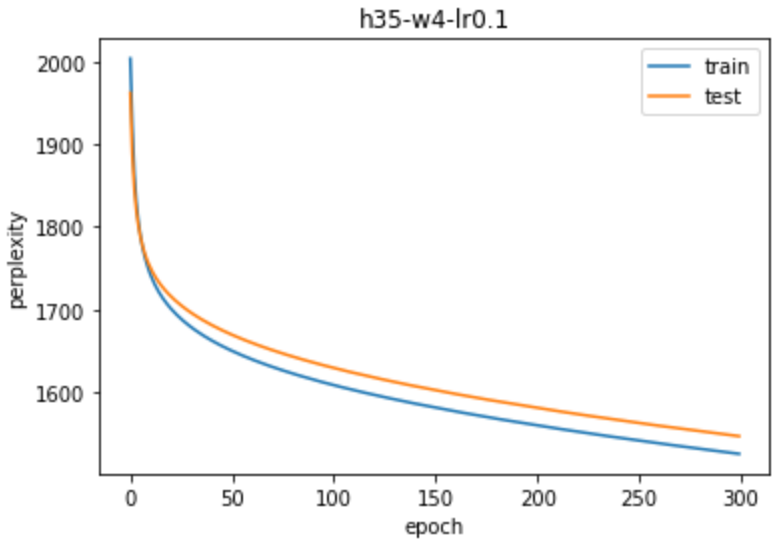
بررسی تأثیر گام آموزش

به سه نمودار زیر با هایپر پارامترهای مکتوب توجه کنید:

Hidden layer neurons number = 35

Window size = 4

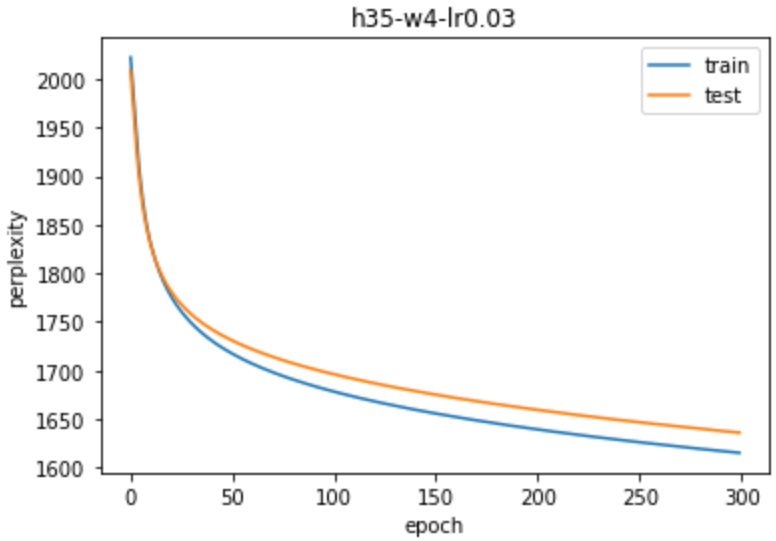
Learning rate = 0.1



Hidden layer neurons number = 35

Window size = 4

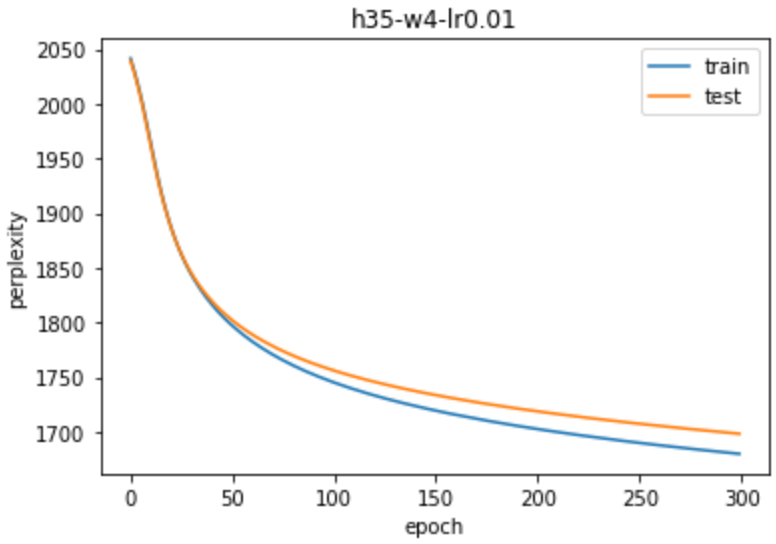
Learning rate = 0.03



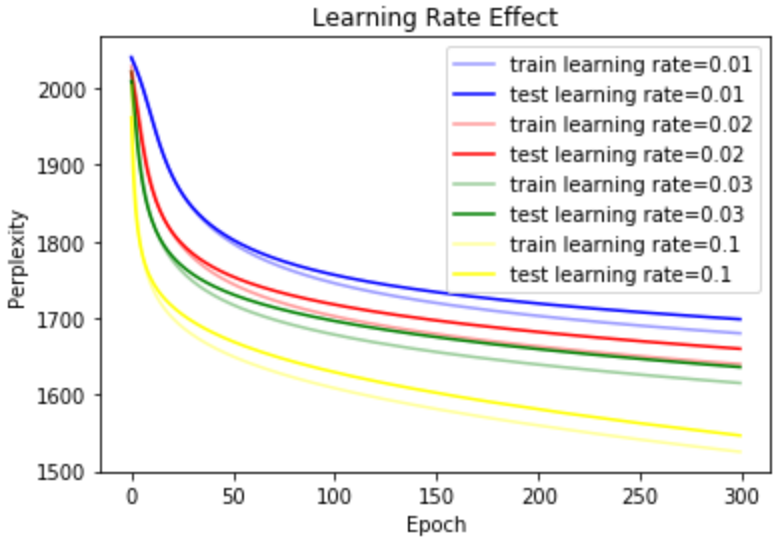
Hidden layer neurons number = 35

Window size = 4

Learning rate = 0.01



مقایسه‌ی نمودارهای بالا را در نمودار زیر مشاهده می‌کنید :



همانطور که می‌توان مشاهده کرد، هرچه learning rate مقدار بزرگ‌تری دارد، نرخ آموزش شبکه در ابتدا سریع‌تر است و به همین دلیل نیز درانتهای آموزش نیز به نتیجه‌ی بهتری می‌رسد. البته انتظار می‌رود اگر آموزش را ادامه می‌دادیم، مدل با learning rate بالاتر زودتر به نقطه‌ی توقف بهبود می‌رسید چرا که دقت آن از حدی بهتر نمی‌شود. همچنین نکته‌ی قابل توجه دیگر این است که در مدل با learning rate کوچک‌تر اختلاف بین نمودار perplexity برای دادگان train و test به مرور افزایش بیشتری نسبت به این اختلاف در مدل‌های با learning rate بزرگ‌تر دارد. دلیل این امر بیش از اندازه دقیق شدن پارامترها نسبت به دادگان ورودی و درنتیجه افزایش overfit در مدل می‌باشد.

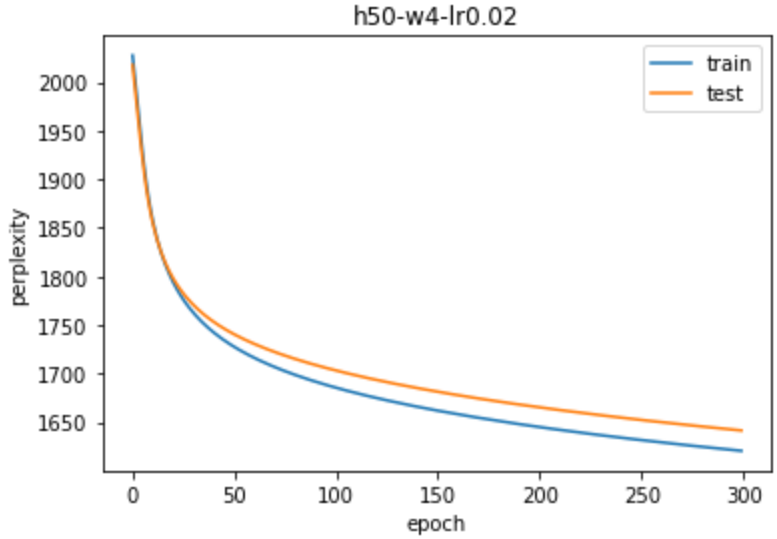
بررسی تأثیر نورون‌های مخفی

به سه نمودار زیر با هایپر پارامترهای مکتوب توجه کنید:

Hidden layer neurons number = 50

Window size = 4

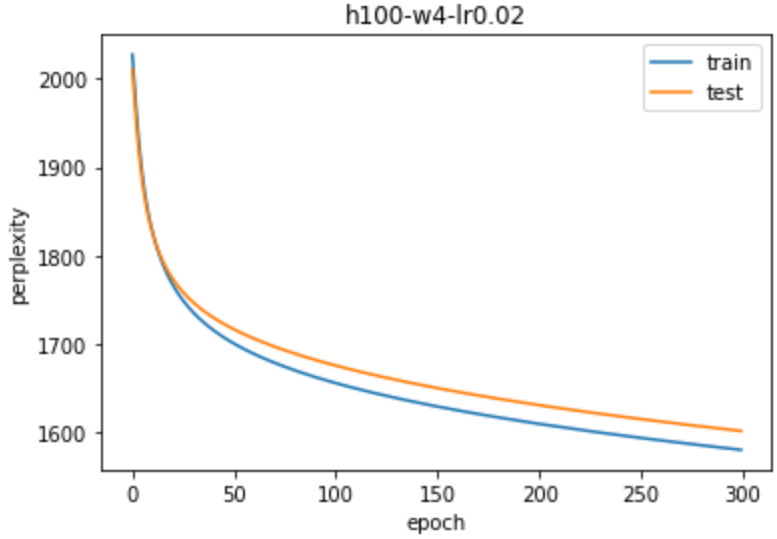
Learning rate = 0.02



Hidden layer neurons number = 100

Window size = 4

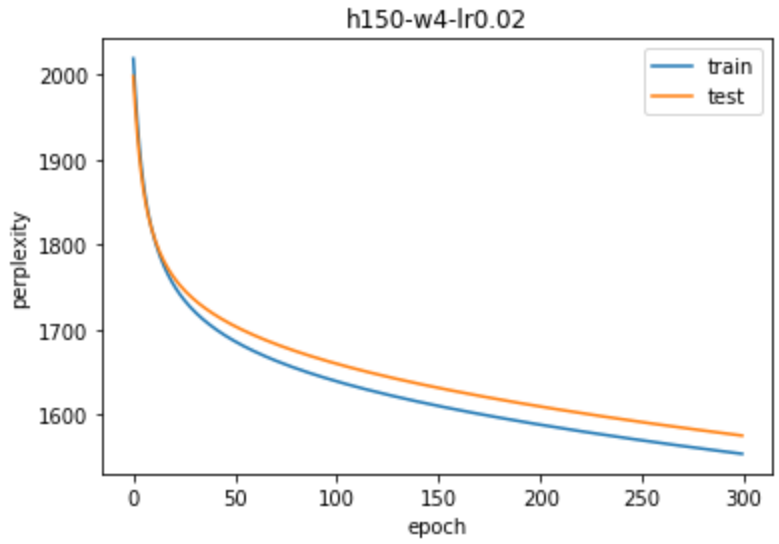
Learning rate = 0.02



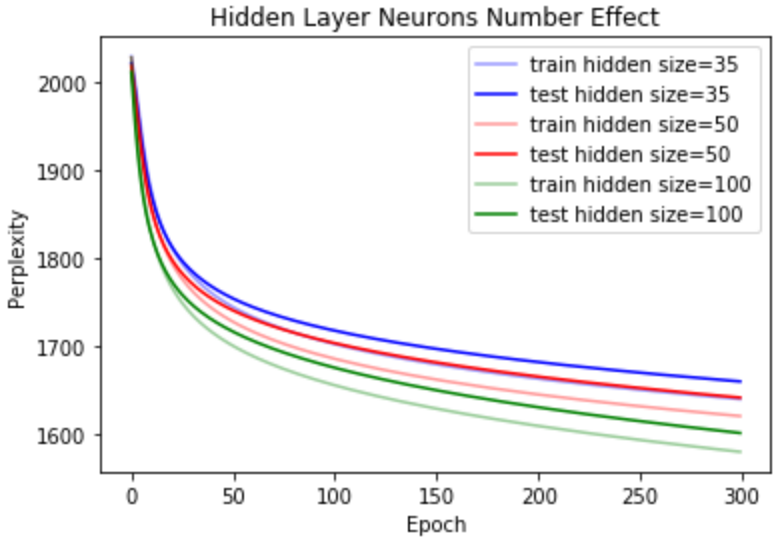
Hidden layer neurons number = 150

Window size = 4

Learning rate = 0.02



مقایسه‌ی نمودارهای بالا را در نمودار زیر مشاهده می‌کنید.



همانطور که مشاهده می‌کنید نرخ نزول Perplexity با اندازه‌ی لایه‌ی مخفی شبکه رابطه‌ی مستقیم دارد و این به آن دلیل است که هرچه تعداد نورون‌های شبکه بیشتر باشد شبکه توانایی بررسی جزئیات بیشتری را دارد و درنتیجه می‌تواند دقت بیشتری بدست آورد و به نتایج بهتری برسد.

بخش دوم) آموزش بردار کلمات در شبکه.

شرح کلی پیاده‌سازی

در این بخش، یک لایه به ابتدای شبکه اضافه کردیم که برداری one-hot به طول تعداد کلمات لغتنامه به عنوان ورودی دریافت کرده و آن را به برداری 50 بعدی به عنوان بردار embedding کلمه تبدیل می‌کند و بدین ترتیب بردارهای embedding را

نیز داخل شبکه آموزش دادیم. ورودی این شبکه اینبار بجای بردار کلمات، یک بردار one-hot یکتا به ازای هر کلمه است. با توجه به اینکه اندازه‌ی این شبکه متناسب با تعداد کلمات لغتنامه بسیار زیاد می‌شود، ناچار شدیم برای انجام پذیر بودن آزمایش تعداد کلمات لغتنامه را کاهش دهیم. بنابراین کلماتی که کمتر از 10 بار تکرار شده بودند را با <UKN> جایگزین کردیم و تعداد لغات را تا 7642 کاهش پیدا کرد. تعداد epochها نیز ۱۲۰ می‌باشد.

نمودار مقادیر Perplexity

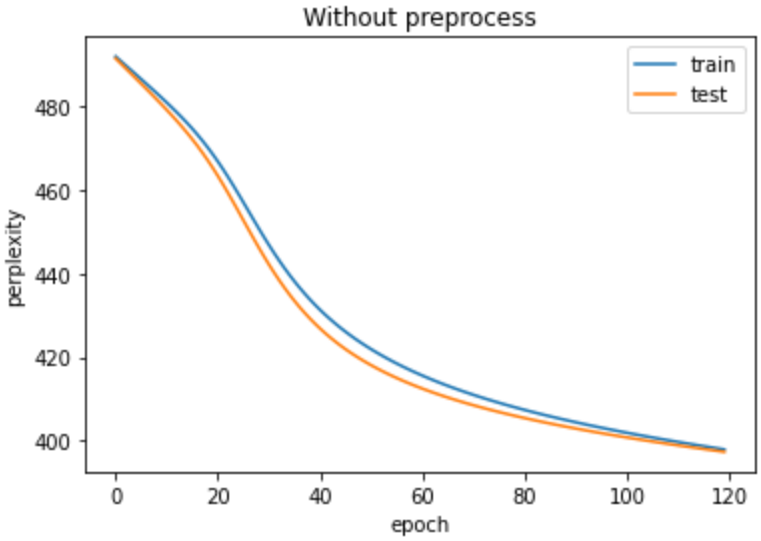
با هایپر پارامترهای زیر، نمودار Perplexity برای iterationهای مختلف به شکل زیر در آمد:

Hidden layer neurons number = 35

Window size = 4

Learning rate = 0.02

Embedding vector size = 50



با مشاهده‌ی نمودار بالا چند نکته حائز اهمیت است. همانطور که مشاهده می‌شود در ابتدا، پیشرفت نمودار به کندی صورت می‌گیرد و این به دلیل زمانی‌است که لازم است تا بردارهای embedding (لایه‌ی پنهان اول) از مقادیر رندم به مقادیر نزدیک به مفهومشان نزدیک شوند. از آن پس است که دیگر بخش‌های شبکه به درستی آموزش می‌بینند. دومین نکته این است که تعداد epochهای طی شده در این آزمایش کمتر از موارد قسمت اول است (به دلیل سرعت اجرای پایین آموزش) پس مقایسه‌ی نرخ آموزش باید با درنظر گرفتن این نکته انجام شود و دیده می‌شود که آموزش در این آزمایش نرخ سریعی دارد. همچنین این شبکه مشخصا پتانسیل بهتر آموزش دیدن را دارن و این به دلیل توانایی تعیین بردارهای embedding متناسب با تسک است و درواقع پارامترهای بیشتری برای آموزش دادن دارد (لایه‌های از پیش آموزش دیده، همواره رشد شبکه را محدودتر می‌کنند). از نکات دیگر کوچک بودن اعداد Perplexity نسبت به آزمایش‌های قبل است که به دلیل آن است که اندازه‌ی لغتنامه و درنتیجه ابعاد ماتریس خروجی را کاهش داده ایم.

فایل‌های جانبی

به همراه این گزارش، یک پوشه به نام Codes ارائه می‌شود که حاوی فایل‌های زیر است :

NLP_CA3_Part0.ipybn : **فایل ژوپیتر کد پروژه که شامل تمام مراحل پیش‌پردازش می‌باشد.**

NLP_CA3_Part1.ipybn : **فایل ژوپیتر کد پروژه که شامل بخش اول می‌باشد.**

NLP_CA3_Part2.ipybn : **فایل ژوپیتر کد پروژه که شامل بخش دوم می‌باشد.**

NLP_CA3_Plots.ipybn : **فایل ژوپیتر کد پروژه که شامل کد ترسیم نمودارهای تلفیقیست.**