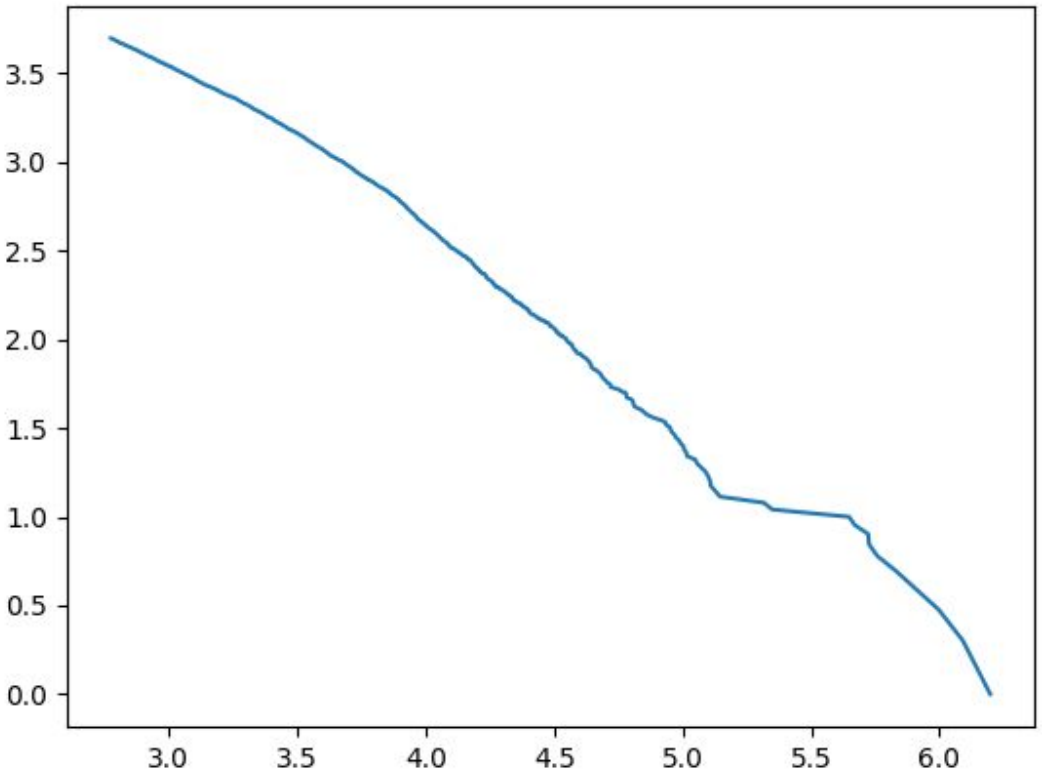
## توضیحات داده:

پیکره شامل ۲بخش که یکی برای آموزش مدل و دیگری برای validation است که داده‌های آموزشی شامل ۱۰۰۰۰۰ خبر و داده validation شامل ۱۵۰۰۰ خبر است.

بعد از چند مرحله نرمال‌سازی و تمیزسازی داده‌های آموزشی این مجموعه شامل ۲۳۶۴۰۷ کلمه متفاوت و ۳۳۵۸۵۶۷۲ توکن است و بعد از محاسبه ۱۰۰۰۰ کلمه پرتکرار می‌توان گفت سهم این کلمات از کلیه توکن‌ها ۳۱۶۱۰۶۳۵ توکن یا به عبارتی حدود ۹۴.۱۲ درصد است و می‌توان گفت با درنظر گرفتن ۵۰۰۰ کلمه پرتکرار این مجموعه داده به میزان خوبی از power law تبعیت می‌کند.




پس از نرمال‌سازی و تمیزسازی داده‌های validation و با استفاده از مقادیر احتمال بدست آمده از مرحله آموزش می‌توانیم evaluation انجام بدیم و روی این مجموعه داده در حالت unigram میانگین WER برای جملات برابر ۱.۸۴ و در حالت bigram برابر ۱.۰۷ و در حالت trigram برابر ۰.۵۷ خواهد شد.

## توضیحات کد:

- فایل normalizer: در این فایل توابع و فرایندهای مرتبط با آماده‌سازی داده برای آموزش آمده‌اند.
  - تابع remove_puncuations: این تابع یک متن به عنوان ورودی دریافت می‌کند و علائم نگارشی(به جز . و ؟) و دیگر علامت‌ها را از متن حذف می‌کند و متن رو برمی‌گرداند.
  - تابع remove_diacritics: این تابع یک متن به عنوان ورودی می‌گیرد و اعراب کلمات مانند فتحه و ضمه و غیره را حذف می‌کند و متن را برمی‌گرداند.
  - تابع remove_emojis: این تابع یک متن به عنوان ورودی می‌گیرد و ایموجی‌های متن را حذف می‌کند و متن را برمی‌گرداند.
  - تابع remove_english_characters: این تابع یک متن را به عنوان ورودی می‌گیرد و حروف انگلیسی در متن را حذف می‌کند و متن را برمی‌گرداند.
  - تابع mask_numbers: این تابع یک متن را به عنوان ورودی می‌گیرد و اعداد ظاهر شده در متن را با عبارت NUM جایگزین می‌کند و متن را برمی‌گرداند.
  - تابع normalizing_training_set: این تابع وظیفه آماده‌سازی داده‌های آموزش را دارد و بعد از خواندن متن چندین گام متن را نرمال‌سازی انجام می‌دهد سپس با توجه به علامت‌های . و ؟ عبارت را بدست می‌آورد و تعداد رخداد کلمات را می‌شمارد و بر مبنای این رخدادها کلمات پرتکرار را می‌یابد و مابقی کلمات را با عبارت UNK جایگزین می‌کند.
  - تابع normalizing_validation_set: این تابع وظیفه آماده‌سازی داده‌های validation را دارد و بعد از انجام چند نرمال‌سازی روی متن کلمات را بدست می‌آوریم و با توجه به کلمات پرتکرار بدست آمده از مرحله قبل کلمات کم‌تکرار با UNK جایگزین می‌کنیم.
- فایل language_model: در این کلاس LanguageModel و توابع مورد استفاده آن قرار دارد.
  - تابع init: این تابع آرگومان‌های ورودی برای ایجاد یک شی از کلاس LanguageModel را به عنوان ورودی می‌گیرد که این آرگومان‌ها آدرس داده‌های آموزشی آماده و کلمات پرتکرار و n_gram و نوع smoothing است.

- تابع train: این تابع با توجه به نوع n_gram و داده‌های آموزشی آماده شده مدل را آموزش را یا به عبارت دیگر رخداد n_gramها را محاسبه کند.
- تابع smoothing: این تابع تعداد رخدادهای یک n_gram خاص و رخدادهای پیش‌نیاز این n_gram را به عنوان ورودی می‌گیرد و مقدار smooth شده احتمال این n_gram را برمی‌گرداند.
- تابع prob_log: این تابع یک مجموعه از کلمات پشت سر هم را به عنوان ورودی می‌گیرد و لگاریتم احتمال آن را با استفاده از رخدادهای بدست آمده در تابع train و تابع smoothing برمی‌گرداند.
- تابع generate: این تابع یک مجموعه از کلمات پشت سر هم را به عنوان ورودی می‌گیرد و با استفاده از تابع log_prob کلمه‌ای با بیشترین احتمال که بعد از این کلمات بیاید را برمی‌گرداند.
- تابع evaluate: این تابع آدرس یک مجموعه داده آماده شده را به عنوان ورودی می‌گیرد و با توجه به تابع generate و الگوریتم shannon جملاتی را تولید می‌کنید و مقدار WER آن‌ها را محاسبه می‌کند.