به نام خدا

۱)

a)

۱- **سطح آوایی**: این سطح ابهام مربوط به بازشناسی گفتار است و هنگامی به وجود می آید که یک عبارت به دو صورت شنیده میشود.

مثال:

۱- او کار ثوابی کرد. معنی: او کاری انجام داد که برای اون پاداش اخروی دارد.

۲- او کار صوابی کرد. معنی: او کار درستی انجام داد.

۲- **سطح نحوی**: این ابهام زمانی اتفاق می افتد که یک جمله به روش های مختلفی تجزیه شود به دلیل اینکه ساختار های نحوی متفاوتی دارد.

مثال: سارا خواهرش را با دوربین مشاهده کرد. این جمله دو معنی میتواند داشته باشد. ۱- سارا خواهرش را به وسیله دوربین مشاهده کرد. ۲- سارا خواهرش را با دوربینی که حمل میکرد مشاهده کرد.

۳- **ابهام معنایی**: این ابهام به این دلیل به وجود می آید که یک کلمه دو یا بیشتر از دو تعریف داشته باشد.

مثال: کلمه "شانه". این کلمه دو معنی دارد. ۱- شانه سر که با آن مو را مرتب میکنند (بُرس). ۲- یکی از اعضای بدن.

۴- **سطح چند عبارتی (discourse / multi-clause)**: این ابهام به دلیل استفاده از ارجاع به ضمیر جهت جلوگیری از تکرار پیش می آید.

مثال: رستم سوار بر اسب شد و دستی بر سرش کشید. در این جمله نمیدانیم که منظور از سرش، سر خود رستم است یا سر اسب.

b)

Text summarization:

تکنیکی برای خلاصه و کوتاه کردن متن های طولانی میباشد. در این تکنیک اندازه متن نسبت به حالت اولیه کاهش میابد اما اطلاعات مفید و کلیدی حفظ میشود.

Entity linking:

قبل از اینکه به سراغ Named Entity Likning یا همان Entity Linking برویم توضیح مختصری راجع به information extraction میدهیم که به فهم ما کمک میکند.

Information extraction: استخراج اطلاعات ساختار یافته به صورت اتوماتیک از داکیومنت های ساختار نیافته. این عمل از سه sub-tasks تشکیل شده است:

1. Named Entity Recognition (NER)
2. Named Entity Linking (NEL)
3. Relation Extraction

Named Entity، موجودیت های دنیای واقعی مانند انسان، زمان و ... است. NER در واقع این موجودیت ها را شناسایی میکند و در دسته های از قبل تعریف شده قرار میدهد. برای مثال خروجی "ثمین حیدریان" در این تسک "انسان" است اما اینکه در مورد کدام ثمین حیدریان صحبت میشود توسط NER مشخص نمیشود.

این کار توسط NEL انجام میشود. NEL هر یک از این موجودیت های شناسایی شده در متن را به یک موجودیت متناظر در پایگاه اطلاعات متصل میکند. این پایگاه اطلاعات میتواند از جایی مانند Wikipedia استخراج شود. برای مثال DBPedia یکی از این پایگاه ها میباشد.

Machine Translation:

یک تسک اتوماتیک که یک زبان طبیعی را به زبان طبیعی دیگر ترجمه میکند. زبان طبیعی ترجمه شده باید روان و سلیس باشد و همچنین معنای زبان طبیعی ورودی را حفظ کند.

c)

الگوریتم یک رشته ورودی که حروف در آن به یک دیگر چسبیده اند را میگیرد و کلمه را به عنوان خروجی برمیگرداند. این الگوریتم به صورت حریصانه عمل میکند. یعنی همیشه بزرگترین کلمه موجود در دیکشنری را برمیگرداند.

شبه کد این الگوریتم به صورت زیر میباشد:

۱- پوینتر را در ابتدای رشته قرار بده.

۲- بلندترین کلمه ی موجود در دیکشنری که با رشته در مکان فعلی پوینتر تطبیق پیدا میکند را پیدا کن.
۳- لغت را در یک ساختار مانند لسیت ذخیره کن.
۴- پوینتر را به مکان بعد از کلمه پیدا شده حرکت بده.
۵- تا زمانی که به انتهای رشته نرسیدی به مرحله دو بازگرد.
۶- لیست کلمات را به عنوان خروجی برگردان.

مثال:

ورودی الگوریتم: sheriftedthewood

خروجی الگوریتم: sherif ted the wood

حالت مدنظر ما: she rifted the wood

کاربرد الگوریتم: برای توکنایز کردن متون زبانی هایی مانند زبان چینی قابل استفاده است.

d)

توضیح lemmatization: هدف این کار کاهش کلمه به شکل پایه یا همان ریشه است. همچنین مترادف های کلمه به یک شکل یکسان یکپارچه میشوند (برای مثال "best" به "good" تبدیل میشود). خروجی یک متن بعد از lemmatization کلمات با معنا و موجود در دیکشنری است.

مثال:

- آسمان ---> آسمان

- رفته ام ---> رو یا رفت
- می روم ---> رو یا رفت

- ❖ خورده اند ---> خور یا خورد
- ❖ میخوری ---> خور یا خورد

توضیح stemming: با این عمل با هدف حذف وندها انجام میشود (چه پیشوند چه پسوند). خروجی یک متن بعد از ریشه یابی یا stemming، لزوما کلمات با معنایی که در دیکشنری موجود هستند، نیست اما پیشوندها و پسوندهای آن حذف شده و در نهایت به ساده ترین حالت در می آید.

مثال:

- آسمان ---> آسم

- رفته ام ---> رفت
- می روم ---> رو

- خورده اند ---> خورد
- میخوری ---> خور

۲)

a) ^[A-Z].*f$

[A-Z]^: رشته با یکی از کاراکتر های بزرگ A تا Z شروع شود.

. : یعنی هر کاراکتری مجاز است.

*: 0 یا بیشتر از آن کاراکتر ما قبل مجاز است.

f$: رشته با کاراکتر f تمام شود.

b) .*4.*4.*4.*

c) ^[13579].*[a-z].*[02468]$