به نام خدا

نام دانشجو: متین مژگانی **نام درس: پردازش زبان های طبیعی**

موضوع : تشخیص شایعه با رویکرد ELKP

مسئله اصلی چیست و چرا مهم است؟

در دنیای امروز، شبکه‌های اجتماعی با سرعتی باورنکردنی اطلاعات را پخش می‌کنند و در این میان، "انتشار اخبار جعلی و شایعات" به یک معضل اجتماعی جدی تبدیل شده است. آمارهای نگران‌کننده‌ای وجود دارد که نشان می‌دهد اکثر بزرگسالان با اطلاعات فریبنده در پلتفرم‌های اجتماعی مواجه شده‌اند. این مسئله فقط یک بحث تئوری نیست؛ دیدیم که مثلاً در دوران کرونا، شایعات چطور باعث ترس عمومی و هدر رفتن منابع پزشکی شد. پس، ساختن یک سیستم خودکار که بتواند شایعه را تشخیص دهد، یک نیاز حیاتی است. اما چرا سیستم‌های فعلی خوب کار نمی‌کنند؟ مشکل اینجاست که روش‌های قدیمی (مثل شبکه‌های عصبی معمولی یا روش‌های آماری) توانایی درک پیچیدگی‌های متن را ندارند و نمی‌توانند شایعاتی که ساختار عجیب یا دوپهلو دارند را درست تحلیل کنند. از طرف دیگر، مدل‌های زبانی بزرگLLM) ها (که اخیراً مد شده‌اند، با اینکه فهم زبانی بالایی دارند، اما باز هم در "تطبیق مستقیم" با این وظیفه خاص مشکل دارند. دلیلش این است که این مدل‌ها "دانش تخصصی" یا "کانتکست" لازم برای قضاوت درباره یک شایعه خاص را ندارند. یک شایعه ممکن است خیلی خوب و رسمی نوشته شده باشد و مدل را گول بزند.بنابراین، مسئله اصلی این مقاله این است: چطور می‌توانیم مدل‌های زبانی بزرگ را طوری مجهز کنیم که علاوه بر فهم زبان، "دانش و فکت" لازم برای تشخیص شایعه را هم داشته باشند و در برابر کمبود داده‌های آموزشی مقاوم باشند؟

ورودی‌ها و خروجی‌های مدل/سیستم چیست؟

سیستم پیشنهادی در این مقاله که ELKP نام دارد، دقیقاً مثل یک ماشین حقیقت‌سنج عمل می‌کند:

ورودی :(Input) ورودی سیستم، یک "متن ادعا (Claim) "است که از شبکه‌های اجتماعی (مثل توییتر) برداشته شده. برای مثال: «لاکی چارمز حاوی مقدار ناامنی از ترکیب پاک‌کننده تری‌سدیم فسفات است.»

خروجی :(Output) خروجی سیستم، یک برچسب (Label) است که وضعیت صحت آن ادعا را مشخص می‌کند. بسته به نوع دیتاست، این خروجی می‌تواند تشخیص شایعه بودن یا نبودن، و یا دسته‌بندی دقیق‌تر مثل "شایعه واقعی(True Rumor) "، "شایعه دروغ(False Rumor) "، "غیر شایعه (Non-rumor) "و "تایید نشده (Unverified) "باشد.

داده مورد استفاده (نوع، منبع، اندازه)

برای اینکه نویسندگان مطمئن شوند روششان واقعاً در دنیای واقعی کار می‌کند و نه فقط روی کاغذ، از سه دیتاست مشهور و واقعی شبکه‌های اجتماعی استفاده کرده‌اند. این داده‌ها شبیه‌سازی شده نیستند و مستقیماً از توییتر جمع‌آوری شده‌اند:

دیتاست‌های Twitter15 و Twitter16 :

این دو مجموعه داده شامل توییت‌هایی هستند که با چهار برچسب دسته‌بندی شده‌اند: غیر شایعه(N) ، شایعه دروغ(F) ، شایعه واقعی (T) و تایید نشده .

دیتاستPHEME :

این دیتاست مربوط به پنج رویداد خبری فوری (Breaking News) است.

برچسب‌ها در اینجا دو حالته هستند: درست (True) و غلط(False) .

روش پیشنهادی مقاله به زبان ساده(ELKP Framework)

نویسندگان این مقاله برای حل مشکلاتی که در بخش قبل گفتیم، یک چارچوب جدید به نامELKP یا (Knowledge-Powered Prompting) طراحی کرده‌اند. اگر بخواهیم خیلی ساده بگوییم، این سیستم به جای اینکه فقط متن را بخواند، مثل یک کارآگاه ابتدا درباره "بازیگران اصلی" آن متن تحقیق می‌کند، اطلاعات اضافی جمع می‌کند و بعد قضاوت می‌کند.

این سیستم چهار مرحله اصلی دارد که در ادامه با جزئیات بررسی می‌کنیم:

مرحله اول: ساختن سوالات اکتشافی (Exploration Prompt Construction) اولین قدم این است که بفهمیم این شایعه اصلاً راجع به چیست.

تشخیص موجودیت‌ها :(NER) سیستم ابتدا متن ادعا را می‌گیرد و با استفاده از ابزارSpaCy ، "موجودیت‌ها " (مثل اسم افراد، مکان‌ها یا سازمان‌ها) را بیرون می‌کشد. مثلاً اگر متن درباره "تری‌سدیم فسفات در غلات" باشد، سیستم متوجه می‌شود که "تری‌سدیم فسفات" کلمه کلیدی است.

طراحی سوال: حالا سیستم نباید خشک و خالی عمل کند. نویسندگان از یک سری قالب‌های آماده (Template) استفاده می‌کنند تا یک "سوال اکتشافی" بسازند. مثلاً می‌پرسند: «ما چه چیزی باید درباره [موجودیت] بدانیم؟».

ساخت پرامپت: این سوال تبدیل به یک پرامپت می‌شود که جاهای خالی (MASK) دارد. هدف این است که مدل مجبور شود این جاهای خالی را با اطلاعات مفید پر کند. مثلاً: «ما باید [MASK] [MASK] درباره تری‌سدیم فسفات بدانیم».

مرحله دوم: ترکیب دانش و پرامپت (Knowledge-Prompt Combination) این مرحله هوشمندانه‌ترین بخش کار است. اگر ما همین‌طوری اطلاعات ویکی‌پدیا را به مدل بدهیم، ممکن است پر از نویز و اطلاعات به‌دردنخور باشد. پس سیستم اینجا یک فیلتر می‌گذارد:

تزریق دانش: سیستم به پایگاه دانش (Knowledge Base) وصل می‌شود و اطلاعات مربوط به آن موجودیت را می‌گیرد (مثلاً فرمول شیمیایی و خواص تری‌سدیم فسفات).

فیلتر کردن نویز: برای اینکه اطلاعات پرت و پلا وارد تصمیم‌گیری نشود، سیستم از مکانیزم "توجه (Attention Mechanism) "استفاده می‌کند. این مکانیزم بررسی می‌کند که کدام بخش از دانش بیرونی با متن ادعا "ارتباط معنایی" دارد.

نتیجه این می‌شود که ما یک "متن غنی‌شده" داریم که هم ادعای اصلی را دارد و هم فکت‌های تایید شده‌ای که مستقیماً به آن ادعا ربط دارند.

مرحله سوم: مدل زبانی بزرگ (LLM Fine-tuning) حالا که داده‌های تمیز و غنی آماده شد، نوبت مغز اصلی سیستم است:

نویسندگان از مدل LLaMA استفاده کرده‌اند. برخلاف مدل‌های قدیمی که نیاز به آموزش سنگین داشتند، اینجا از روش "تنظیم دقیق پرامپت " (Prompt Tuning)استفاده می‌شود.

اطلاعات غنی‌شده وارد مدل می‌شود و مدل سعی می‌کند با استفاده از دانش قبلی خودش و دانش جدیدی که به آن تزریق شده، متن را تحلیل کند.

مرحله چهارم: بهینه‌سازی مشترک (Joint Optimization) این قسمت فنی ماجراست که باعث می‌شود مدل همزمان دو کار را یاد بگیرد. سیستم دو هدف (Loss Function) دارد که با هم جمع می‌شوند:

پر کردن جاهای خالی :(MLM Loss) مدل سعی می‌کند کلمات حذف شده در پرامپت را حدس بزند. این کار باعث می‌شود مدل "فهم زبانی" و "دانش متنی" خود را تقویت کند.

تشخیص شایعه :(Classification Loss) همزمان، یک شبکه عصبی کوچک به انتهای مدل وصل است که سعی می‌کند پیش‌بینی کند آیا خبر درست است یا غلط.

شبه‌کد (الگوریتم) به زبان ساده: اگر بخواهیم روند کار (الگوریتم ۱ در مقاله) را خلاصه کنیم، اینطور می‌شود:

برای هر ادعا در مجموعه آموزشی:

موجودیت‌های مهم را پیدا کن.(NER)

یک پرامپت سوالی بساز(Prompts) .

دانش مرتبط را از دیتابیس خارجی پیدا کن.

دانش را فیلتر کن و با پرامپت ترکیب کن(Knowledge Injection) .

متن ترکیبی را به مدل LLaMA بده.

همزمان خطا (Loss) را برای "پر کردن جاهای خالی" و "تشخیص کلاس شایعه" محاسبه کن.

مدل را آپدیت کن تا هر دو خطا کم شود.

نتایج اصلی، مقایسه و تحلیل

نویسندگان برای اثبات کارایی مدل خود، آن را با طیف وسیعی از مدل‌ها مقایسه کردند: از روش‌های کلاسیک مثل SVM و درخت تصمیم گرفته تا مدل‌های عمیق (CNN, GRU) و حتی مدل‌های پیشرفته گرافی (TextGCN) و زبانی(BERT) .

برتری نسبت به رقبا: نتایج نشان داد که روش ELKP در هر سه دیتاست عملکرد بهتری داشته است. به طور خاص، در دیتاست‌های Twitter15 وTwitter16 ، این مدل توانست بهترین رقیب خود را به ترتیب ۱.۶٪ و ۲.۴٪ شکست دهد. در دیتاست PHEME اختلاف حتی فاحش‌تر بود و این مدل حداقل ۴.۲٪ بهتر از روش‌های استخراج خودکار عمل کرد.

چرا LLaMA انتخاب شد؟ مقاله مقایسه‌ای بین مدل‌های زبانی مختلف BERT, RoBERTa, BART, XLNet و LLaMA انجام داد. نتایج نشان داد که مدل LLaMA به دلیل توانایی بالاتر در درک متون پیچیده و استدلال معنایی، بهترین گزینه برای هسته مرکزی این سیستم است.

تاثیر اجزای مختلف :(Ablation Study) نویسندگان برای اینکه مطمئن شوند همه بخش‌های سیستم لازم هستند، تست‌های جالبی انجام دادند:

وقتی فیلتر دانش را حذف کردند (یعنی دانش ویکی‌پدیا را بدون فیلتر به مدل دادند)، دقت حدود ۳٪ افت کرد. این ثابت می‌کند که اگر نویز را حذف نکنیم، تزریق دانش می‌تواند نتیجه عکس بدهد.

وقتی مدل زبانی بزرگ (LLM) را حذف کردند و جای آن یک شبکه ساده گذاشتند، افت عملکرد بسیار شدید بود که نقش حیاتی مدل زبانی را نشان می‌دهد.

همچنین مشخص شد که طول بهینه برای جاهای خالی (MASK) در پرامپت، ۱۰ توکن است. کمتر از این اطلاعات کافی نمی‌دهد و بیشتر از این باعث گیج شدن مدل می‌شود.

انتخاب ابزار :NER برای بخش تشخیص موجودیت، ابزارهای مختلفی مثل Stanford NER و NLTK تست شدند، اما نهایتاً SpaCy به دلیل تعادل بین سرعت، دقت و پشتیبانی چندزبانه انتخاب شد.

محدودیت‌ها و کارهای آینده

تعمیم‌پذیری :(Generalizability) یکی از اهداف آینده این است که ببینند آیا این چارچوب (ترکیب دانش و پرامپت) روی سایر وظایف NLP که نیاز به درک معنایی عمیق دارند هم جواب می‌دهد یا خیر.

تفسیرپذیری :(Interpretability) مدل فعلی خوب کار می‌کند، اما هنوز دقیقاً مشخص نیست که مدل "چگونه" استدلال می‌کند. کارهای آینده قرار است روی این تمرکز کنند که فرایند استدلال معنایی مدل شفاف‌تر و قابل‌فهم‌تر شود.

پویایی دانش: بهبود نحوه استفاده از دانش و مدل‌سازی دینامیک (پویا) برای تطبیق با شایعات جدید و در لحظه، از دیگر اهداف پژوهش‌های بعدی است.