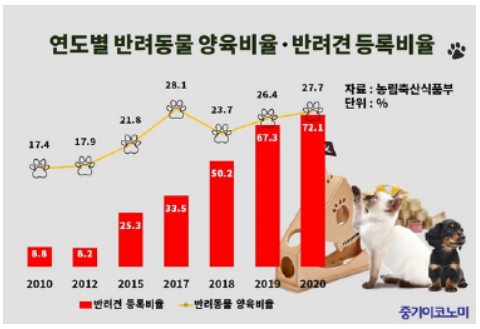
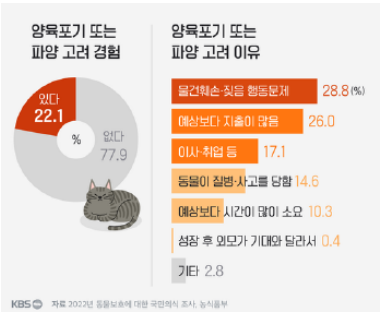
2113862 김민경 텍스트마이닝과 자연어처리 기말과제

주제: 펫보험(반려견 보험) 약관 텍스트 분석을 통한 RAG 기반 펫보험 추천 파이프라인 설계

1. 분석 주제의 필요성과 사회적 효용성





[그림 1] 연도별 반려동물 양육비율 반려견 등록비율[1] / [그림 2] 양육포기 또는 파양 고려 이유[2]

[그림 3] 반려동물(펫보험) 계약건수 및 원수보험료 추이[3]

현대 사회에서 반려동물, 그 중에서도 특히 반려견을 양육하는 가구가 지속적으로 증가하는 추세에 있다. 하지만 반려견 양육에 따른 비용, 특히 질병 치료로 인한 의료비는 보호자에게 큰 경제적 부담을 주며 양육 포기 혹은 파양을 고려하게 하는 원인이 되기도 한다. 이에 따라 펫보험에 대한 관심과 가입 건수는 매년 증가하는 추세이다.
그러나 국내 펫보험 상품은 약관 문서가 매우 복잡하고 보장 범위·조건·절차 등이 상세히 기술되어 있어 소비자가 스스로 자신의 상황에 적합한 상품을 비교, 선택하기 어렵다. 또 기존의 펫보험 비교 서비스는 대개 미리 정의된 한정된 카테고리로만의 비교를 제공하며, 세부적인 절차나 조건을 한눈에 파악하기 힘들다는 한계가 있다.

---

[1] 638만 가구에서 반려동물 860만 마리 키운다 [중기이코노미]
[2] 반려동물 양육비도 올랐다…한달에 얼마나 드나? [KBS]
[3] 펫보험' 영업 강화 나선 손보업계…수익 확대 효자될까 [시사저널e]

이에 따라 펫보험 약관을 텍스트 분석 기반으로 자동 비교할 수 있는 시스템을 구축하고, 챗봇을 통해 직관적인 정보를 제공하고자 한다. 이를 통해 반려동물 보호자의 펫보험에 대한한 정보 접근성을 높이고, 보다 합리적인 의사결정을 지원하고자 한다.

2. 분석 및 구현 목표

텍스트 마이닝 및 자연어처리(NLP) 기법을 활용하여 국내 주요 반려견 보험 약관 문서를 분석하고, RAG(Retrieval-Augmented Generation) 기반 검색 및 추천 시스템을 구현함으로써 보호자에게 개인 맞춤형 보험 비교 정보를 제공하는 방안을 제안한다. 특히 다음과 같은 요소들에 주안점을 두어 방법론을 설계해 실험하였다.

- 보험 약관 텍스트에서 보장 범위, 보장 조건, 보장 절차 등 유형별 핵심 정보를 효과적으로 추출하고 분류할 수 있는가?
- 벡터 유사도 기반 검색에 질의 유형 분류와 키워드 매칭을 결합함으로써 검색 정확도를 향상시킬 수 있는가?
- 구현된 시스템이 보호자의 실제 질의에 대해 유의미한 비교 분석 결과 및 최종 추천을 제공할 수 있는가?

이에 본 프로젝트는 복잡한 보험 약관 문서를 이해하기 어려운 사용자 문제를 해결할 뿐만 아니라, 공공서비스·정책 문서 비교 등 다양한 도메인에도 확장 가능한 텍스트 분석 및 RAG 검색 강화 방법론을 제시한다.

3. 분석 및 구현 방법

3.1 텍스트 데이터 수집 및 전처리

데이터 수집 국내 주요 반려견 보험을 취급하는 대표 3개 보험사(삼성화재, 메리츠화재, 한화보험)의 공개된 반려견 보험 약관 PDF 문서를 수집하였다. PDF 문서는 회사별 약관 조항이 구조화되어 있었으나, 보다 정확한 텍스트 마이닝 처리 방법론 적용을 위해 텍스트 추출 후 추가 전처리 작업을 수행하였다.

- PDF 파싱: PyMuPDF 등 라이브러리를 사용해 약관 PDF에서 텍스트를 추출.
- 조항 단위 Chunking: 약관 문서를 논리적 조항별로 분할하여 각각을 하나의 "chunk"로 정의/ 문단 경계, 조항 번호 등을 이용해 분할
- 문장 분할 및 형태소 분석: KoNLPy의 Okt 등을 활용해 문장을 분리하고 한국어 형태소 분석을 수행하여 명사 및 핵심 형태소를 추출
- 불용어 제거: 일반 불용어뿐 아니라 도메인 불용어(예: "본 약관", "회사", "보험" 등 반복 빈도가 높으나 유의미도 낮은 단어) 사전 구축 후 제거

3.2 TF-IDF 기반 키워드 추출

TF-IDF 기반 키워드 추출 각 chunk별로 TF-IDF를 계산하여 상위 빈출명사 및 핵심어를 추출하였다. 이를 통해 각 조항의 주요 주제를 파악하고, 이후 클러스터링 및 카테고리 분류에 활용할 키워드 집합을 확보했다.

3.3 키워드 클러스터링 및 유형 매핑

질의 유형 카테고리 정의 보장 정보를 세부적으로 탐색하기 위해 미리 정의된 카테고리 3가지를 설정하였고: 보장 범위(예: 입원·외래·응급 등), 보장 조건(예: 보장 한도·보상금액 등), 보장 절차(예: 보험금 청구·증빙 서류 등) 각 카테고리별로 대표 키워드 사전 목록을 구축하였다.

그리고 다음 절차에 따라 앞서 추출한 TF-IDF 키워드들 및 보장 범위/조건/절차 카테고리별 키워드들을 임베딩, TF-IDF 키워드들에 대해 클러스터링을 진행, 각 클러스터별로 의미적 유사도(코사인 유사도)를 계산해 보장 범위/조건/절차 카테고리 할당 과정을 수행했다.

- OpenAI 임베딩: TF-IDF로 추출된 키워드를 OpenAI embedding API로 벡터화
- 클러스터링: K-means 알고리즘을 사용하되, 실루엣 계수 등의 지표로 최적의 군집 수를 결정. 각 군집 내 키워드는 의미적으로 유사한 그룹을 형성 (Appendix [그래프 1] 참고)
- 카테고리 매핑: 각 클러스터의 대표 벡터(평균 벡터)와 미리 정의한 카테고리별 대표 키워드 임베딩 간 유사도를 계산하여 해당 클러스터가 어느 카테고리에 속할 가능성이 높은지 판단. 이로써 키워드 클러스터에 보장 범위·조건·절차 라벨을 할당

3.4 보험 문건 약관 Chunk별 카테고리 비율 산정

하나의 조항(chunk)에는 여러 유형의 정보가 혼합될 수 있으므로, 각 chunk 내 TF-IDF 기반 키워드 및 해당 키워드 클러스터 라벨을 활용해 보장 범위·조건·절차에 해당하는 키워드 출현 비율을 계산했다. 이를 통해 chunk별로 각 카테고리 문맥 비율을 산정하였다.

3.5 요약 임베딩 및 벡터 DB 구성

RAG 적용 시 각각의 보험 약관을 함축적으로 이해하고 정보를 검색할 수 있도록 다음 과정에 따라 각 보험 약관을 요약하여 임베딩하였다. 답변 시에는 보험 약관 원문을 참고하여 답변할 수 있도록 메타데이터로 원문을 함께 Pinecone 벡터 DB에 저장하였다.

- Chunk 요약: 각 chunk의 원문을 OpenAI API를 통해 간결히 요약하여 의미론적 핵심 정보만 추출
- 요약 임베딩: 요약문을 OpenAI embedding API로 벡터화
- 벡터 DB 업로드: Pinecone DB를 활용해 임베딩 벡터와 메타데이터(회사명, chunk 원문, 요약문, TF-IDF 키워드, 카테고리 비율 정보)를 함께 저장

3.6 질의 전처리 및 검색 강화

RAG 기반 보험 비교검색 사용자의 질문 내용이 보험 약관과 의미적 유사도가 떨어지거나 동떨어져 있을 수 있는 상황을 대비하기 위해 질의 내용과 보험 약관 텍스트 간 의미 격차를 줄이고자 다음 과정을 수행하였다.

- 대표 키워드 추출: 벡터 DB의 요약 임베딩들을 다시 클러스터링하여 각 군집에 대표 키워드(GPT API로 추출)를 지정
- 질의 확장: 사용자 입력 질의에 대표 키워드를 결합(concat)하여 OpenAI embedding API로 임베딩
- 검색 가중치 조합: RAG 검색 시 3가지 점수 요소를 결합: 벡터 유사도(질의 확장 임베딩과 chunk 요약 임베딩 사이의 유사도), 카테고리 유사도(사용자 질의에 대한 카테고리 예측 결과(앞선 방법과 동일하게 미리 정의된 카테고리별 키워드들의 평균 벡터와 비교하여 도출)에 따른 chunk의 카테고리 비율), 키워드 포함 점수(질의에 포함된 명사형 핵심어가 chunk 내 TF-IDF 키워드에 얼마나 포함되는지)
    - 예) W_vec * vec + W_cat * cat + W_key *key 조합.
    - (W_vec + W_cat + W_key = 1)

3.7 RAG 기반 응답 생성 및 비교 분석 구현

질문자가 질문에 대한 답변을 한눈에 알기 쉽게 보고 비교 분석할 수 있도록 각 회사별로 질의와 관련된 내용 항목을 표로 정리하여 나타내었다. 질의 - 표 응답 형식의 UI는 GTP API와 streamlit을 이용해 구현하였다.

- 유사도 상위 N개 chunk(회사별 상위 N) 정보를 표 형식으로 정리: 회사별로 보장 범위·조건·절차 측면에서의 주요 정보 발췌
- GPT 기반 응답: 정리된 정보를 바탕으로 GPT API 호출하여 각 보험사별 비교 요약 및 최종 추천을 생성
- Streamlit 시연: 인터페이스 상에서 사용자 질의 입력, 검색 및 비교 분석 결과 시각화, 추천 결과 제공 과정을 구현

4. 결과 및 해석

4.1 텍스트 분석 결과 검증

- 요약문 벡터 클러스터링 결과: Vector DB에 업로드된 요약문 벡터들로 클러스터 응집도 및 카테고리 매핑 적절성 검토한 결과 비교적 클러스터들이 깔끔하게 군집화되었음을 확인하였다. 이는 확장된 질의문에 덧붙여지는 키워드들이 각 군집을 대표적으로 분명하게 나타낼 수 있음을 의미한다. (Appendix [그래프 2] 참고)

- 각 회사별로 약관별 카테고리 분포: 약관 조항 내 정보 혼합 정도 파악한 결과 (특정 조항이 보장 범위·조건 정보를 동시에 포함하는 비율) 각 보험사별로 약관 조항 내 카테고리별 비율이 유사하게 나타났다. 이는 회사별로 비교할 약관을 retrieve할 시에 카테고리 유사도 점수 공정성의 기반이 되는 수치적 결과이다. (Appendix [그래프 3] 참고)
- GPT 기반 추천의 타당성: 예시 시나리오(예: 특정 연령·질병 이력 반려견 보호자)에 대한 표 형식 분석 결과와 그에 따른 추천 결과를 몇 개의 예시 질문들을 통해 검토한 결과 보다 용이하고 보험 약관 검색이 가능함을 확인하였다.

4.2 한계점

- 데이터 한계: 공개된 약관 PDF만 분석 대상. 실제 보험 상품별 세부 조항 변경, 개별 조건 반영 어려움
- 비용 및 성능: OpenAI API 호출 횟수와 비용 고려. 실시간 응답 성능 보장 어려움
- 객관적 성능 평가의 한계 : GPT로 각 보험 약관 chunk별로 질문들을 추출해 실제 RAG 응답 시 해당 질문에 대한 답변이 해당 chunk를 참고하여 답변했는지 여부를 가지고 accuracy를 측정한 결과 키워드 포함 점수(W_key)가 높을수록 accuracy가 높아짐을 확인함. (Appendix [표1] 참고)
  하지만 GPT로 각 보험 약관 chunk별로 질문들을 추출하는 과정에서 질문에 각 조항 약관의 용어들을 포함해 질문들을 구성했을 확인해 다소 편향적인 데이터셋에 따른 결과라고 판단, 성능 평가를 위한 객관적인 QA 데이터셋 확보에 어려움을 겪음

5. 시사점 및 확장 방안

이 프로젝트에 사용된 방법론은 반려견 보험 도메인을 넘어 공공서비스나 정책 문서와 같이 복잡한 구조를 지닌 다양한 문서 비교에도 적용될 수 있다. 특히, 도메인별로 미리 정의된 카테고리와 키워드 클러스터링 방식을 활용한 검색 강화 방법은 유사한 복잡 문서의 핵심 정보를 효과적으로 추출하여 맞춤형 안내 서비스를 제공하는 데 유용하다.

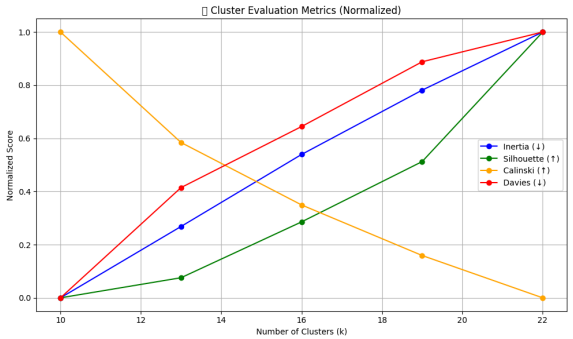
사용자 인터페이스 측면에서는 챗봇 연동이나 음성 질의 지원을 통해 보다 직관적이고 접근성이 높은 서비스 제공이 가능하며, 이러한 인터페이스 개선은 최종 사용자의 경험을 크게 향상시킬 것을 기대할 수 있다.

6. 결론

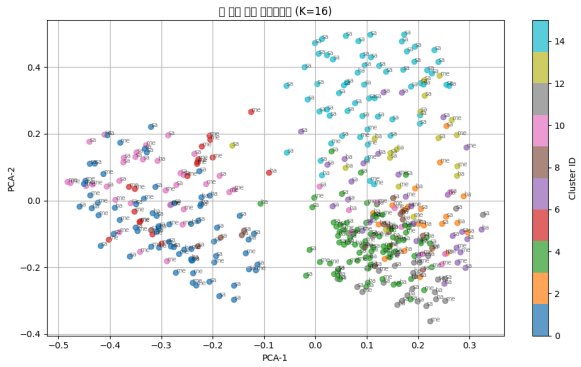
본 프로젝트에서는 반려견 보험 약관의 복잡성을 완화하고 사용자 맞춤형 정보 제공을 위해 텍스트 마이닝·NLP 기법과 RAG 기반 검색 강화 방법론을 제안·구현하였다. 키워드 클러스터링을 통한 카테고리 분류, 요약 임베딩 기반 벡터 DB 구성, 질의 확장 및 가중치 기반 검색 조합이 핵심적인 구현 요소들이다. 구현 결과 해당 RAG 기반 질의-응답 결과는

사용자의 다양한 질의에 대해 회사별 비교 분석 및 추천을 제공하며, 실무적 가치와 확장 가능성을 동시에 지닌다.

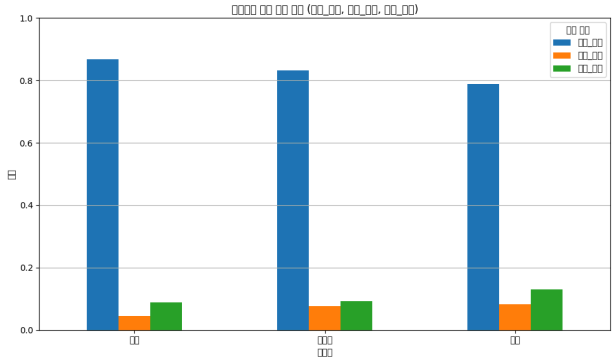
<< Appendix(부록) >>



[그래프1] 최적의 클러스터링 갯수 산정을 위한 실루엣 계수 등의 지표를 확인한 결과



[그래프 2] 요약문 벡터 클러스터링 결과



[그래프 3] 각 회사별로 약관별 카테고리 포함 비율 (파랑 : 보장 범위 / 주황 : 보장 조건 / 초록 : 보장 절차)

W_vec	W_cat	W_key	Accuracy	Hits	Total Queries

0.6	0.1	0.3	0.6258	729	1165
0.6	0.2	0.2	0.6232	726	1165
0.8	0.1	0.1	0.6232	726	1165
0.8	0.0	0.2	0.6146	716	1165
1	0.0	0.0	0.6137	715	1165
0.6	0.3	0.1	0.6026	702	1165
0.8	0.2	0.0	0.594	692	1165
0.6	0.0	0.4	0.5845	681	1165
0.6	0.4	0.0	0.5485	639	1165

[표1] W_vec, W_cat, W_key에 따른 Accuracy 측정 결과

- W_key가 높을수록 더 향상된 Accuracy(데이터셋의 편향성으로 인한 결과라고 추정됨)

- key, cat 둘 중에 하나만 단독 사용 시에는 vec만 사용했을 때보다 다소 낮은 Accuracy

- cat은 key와 함께 보조적으로 사용되었을 때 향상된 Accuracy를 보임