기계학습과 언어처리에 기반한 문자메시지 분류

선주오○, 지명근, 최범휘, 이현아
금오공과대학교, 컴퓨터소프트웨어공학과
qssz1326@naver.com, dldhk97@naver.com, bum4496@naver.com, halee@kumoh.ac.kr

Text Message Classification based on Machine Learning

Juoh Sun○, Myeonggeun Ji, Beomhwi Choi, Hyunah Lee
Kumoh National Institute of Technology, Department of Computer Software Engineering

요 약

휴대전화 메시지로는 결제, 인증번호, 택배, 광고 등의 다양한 문자들이 수신된다. 이 문자들은 서로 섞여 있어 이용자가 찾고자 하는 문자를 찾는 데 어려움이 있다. 본 논문에서는 기계학습과 단어 임베딩을 통해 메시지들을 카테고리로 분류하는 방법을 제안하고, 이를 구현한 안드로이드 앱을 소개한다. 앱에서는 택배, 카드, 인증, 공공기관, 통신사, 대화, 기타의 7개의 분류로 메시지를 분류하며, 자동 분류에서는 수동 태깅한 5802건의 문자메시지를 사용한다. 앱에서는 저장된 문자메시지간 유사도에 기반한 오프라인에서의 자동 분류를 지원하여 개인정보 노출에 대한 거부감이 있는 사용자의 요구를 반영한다.

주제어 : 기계학습, 단어 임베딩, multi-classification, n-gram, tf-idf

1. 서론

최근에는 카카오톡, 라인, 비트윈 등 다양한 무료 메신저가 등장했지만, 여전히 공식적인 문자나, 택배, 카드결제, 광고성 문자 등 다양한 종류의 문자메시지가 휴대전화 문자메시지로 수신되고 있다[1]. 사용자의 요청으로 수신되는 광고나 소속되고 있는 회사나 학교에서 발송한 문자, 택배 배송 문자와 같은 서비스들은 전화번호를 기반으로 정보를 전달하므로 앞으로도 휴대전화 문자메시지 사용은 지속될 것으로 보인다.

휴대전화 메시지 앱의 경우 다양한 문자가 수신된 순서대로 나열되기 때문에 사용자가 원하는 문자를 찾기 어렵다. 수신한 문자는 수동으로 정리해야 하며 수시로 정리하지 않고 방치하면 문자메시지함 관리가 불가능한 상황에 다다르기 쉽다. 이러한 불편을 해결하기 위해 2015년에 SKT통신사에서 '여름'이라는 메시지 분류 어플리케이션을 출시했지만, 2017년 6월 30일부로 서비스를 중단하여 현재는 사용할 수 없다[2]. 문자메시지에서 스팸문자를 분별하는 다양한 연구들[3,4]에서는 문자의 스타일 자질을 이용한 내용 기반 분류나, 워드 임베딩과 딥러닝 기반 분류로 문자메시지를 스팸과 햄으로 이중 분류한다.

근래 카드결제나 택배 문자, 유용한 광고 문자의 양이 급증하고 있으며 이런 문자가 자동으로 분류되면 사용자 편의를 크게 증대시킬 수 있다. 이를 위한 자동 분류는 기존의 스팸과 햄으로 분류하는 이진 분류가 아닌 다중 분류가 필요하며, 시스템의 화면도 다르게 구성해야 한다. 본 논문은 수신된 문자메시지를 분석하여 문자메시지 내용을 기반으로 문자메시지를 여러 카테고리로 분류하고 제시하는 방법을 제안한다.

2. 언어처리에 기반한 문자메시지 분류

2.1. 문자메시지 분류

[1]은 2016년에 전 연령대의 4813명에 대한 문자메시지에 대한 설문을 시행하였으며, 수신된 문자메시지의 종류별 비율을 친목/지인과의 연락 19.2%, 카드 사용 문자 21.8%, 택배 배송 문자 14%, 스팸/스미싱 12.4%, 광고문자 28.9%, 기타 3.6%로 얻었다. 본 연구에서는 2019년 시점에서의 문자메시지 분류를 결정하기 위해 20대 대학생 5명의 문자메시지 5,802건을 수집하여 수동 분류를 시행하였으며 분류 결과에서는 택배 4.3%, 카드 19.2%, 인증 9.7%, 공공기관 3.7%, 통신사 9.7%, 대화 31.1%, 기타 22.3%로 나타났다. 본 논문에서는 이 7개의 분류를 문자메시지 분류로 사용하기로 한다.

2.2. 머신러닝기반 문자메시지 분류

그림 1은 제안하는 시스템의 메시지 분류 처리 과정을 보인다. 문자메시지 수신시 발신번호가 연락처에 있는

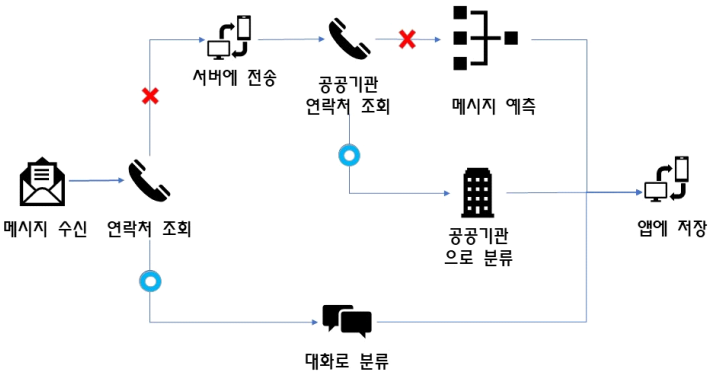

그림 1 시스템의 메시지 분류 진행 단계

번호인 경우는 서버로 송신하지 않고 대화로 분류를 수

행한다. 그렇지 않은 경우에만 서버로 문자메시지를 보내 분류를 수행한다. 서버에서는 공공기관 전화번호를 별도로 관리하여 공공기관인 경우에는 즉시 공공기관으로 분류하고, 이외의 문자메시지에 대해 학습데이터에 기반한 자동 분류를 시행한다.

그림 2는 문자메시지의 예를 보인다. 예에서 볼 수 있듯이 '카드'나 '택배' 등의 명사는 분류에 중요한 정보가 된다. 광고성 문자나 공적인 문자들의 경우 명사가 생략되는 경우가 적어 메시지로부터 명사를 추출해서 분류를 수행한다.

(1) "[Web발신]NH카드 승인 ○○○ 40,000원 체크 07/07 12:00 ㈜티○니시외○○"
(2) "상품을 cj택배 ○○ 편의점에(으로) 보관(전달)하였습니다. 미확인시 연락주세요(CJ택배)_123412341234"
(3) "[Web발신][네○버] 인증번호 [123456]를 입력해주세요."
(4) "[기상청] 04월22일05:45 경북 울진군 동남동쪽 43km 해역 규모4.0 지진발생/낙하물로부터 몸 보호, 진동 멈춘 후 야외 대피하며 여진주의"
(5) "[Web발신][K○알림]데이터박스 2000MB 담기성공. 담은 데이터는 다음 달 말일까지 유지됩니다."

(1)-2 NH, 카드, 승인, 체크,
(2)-2 상품, cj, 택배, 편의점, 보관, 전달, CJ
(3)-2 인증번호
(4)-2 기상청, 경북, 울진, 해역, 지진, 대피, 여진
(5)-2 데이터, 박스

그림 2 문자메시지와 문자메시지의 명사

본 논문에서는 명사 추출을 위해 Okt(Open Korean Text)[5]를 사용한다. 실험에서는 Word2Vec의 CBOW(Continuous Bag of Words)과 Skip-gram을 모두 적용하여 그 결과를 살펴보고, 우수한 방식을 시스템에 적용한다.

학습데이터는 2.1에서 수집한 5,802건 중 3,942개의 문자메시지를 사용한다. 택배, 카드, 인증, 공공기관, 통신사, 대화, 기타의 해당 태그와 벡터화한 학습데이터로 문자메시지 분류 모델을 생성한다.

메시지 분류 모델은 MLP(Multi-Layer Perceptron)을 사용한다. 앞서 Word2Vec을 통해 문자메시지를 200차원 벡터로 변환하였다. 이 값을 이용하여 MLP모델을 통해 문자메시지를 분류한다.

메시지 분류 모델 학습 시 활성화 함수는 ReLU (Rectified Linear Unit) 함수를 사용하였다. ReLU 함수의 경우 양수인 경우에서 Sigmoid 함수가 갖는 단점인 Gradient vanishing 문제를 해결할 수 있으며, 지수함수인 Sigmoid 함수와 달리 계산 복잡성이 낮아 학습 수렴 속도도 6배나 빠른 효과를 얻을 수 있기 때문이다[6].

그림 3에 따르면 학습 epoch이 증가할수록 정확도가 증가하는 것을 볼 수 있으나, 100epochs 이후로는 증가폭이 미미한 것을 볼 수 있다. 따라서 정확도가 90%를 넘는 200epochs으로 학습을 수행했다.

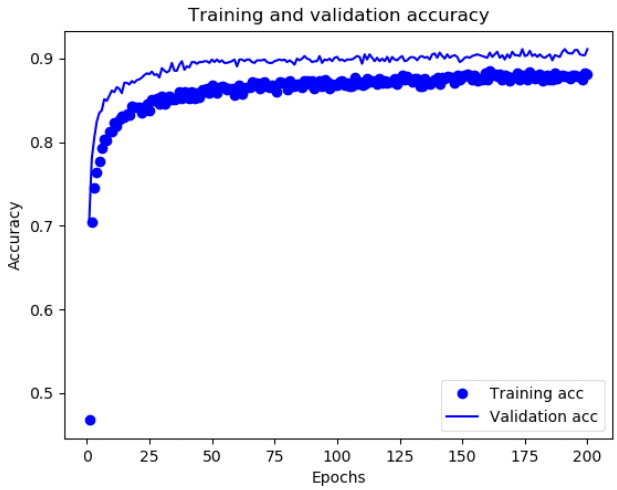

그림 3 학습 Epochs에 따른 정확도 그래프

2.3 오프라인 상황을 위한 유사도기반 분류

2.2의 분류 모델은 사용자들의 문자메시지가 축적될수록 더 좋은 성능을 낼 수 있다. 이를 위해서는 사용자들의 문자메시지를 수집해야 하지만, 이 방식은 개인정보 노출에 대한 사용자의 우려나 부담이 큰 문제가 있다. 이를 해결하기 위해 스마트폰 단말에서 학습을 수행하려면 Word2Vec과 Tensorflow, ML.net 등이 필요하지만 사용이 원활하지 않다.

제안하는 시스템에서는 개인정보 노출에 대한 거부감이 있는 사용자들을 위해 오프라인에서의 분류를 제공한다. 시스템의 7개 분류의 문자메시지 집합을 C_k(1 <= k <= 7), 신규 수신된 문자메시지를 s로 정의하는 경우, C_k에 포함된 문자메시지와 s의 유사도를 사용하면 s의 분류를 결정할 수 있다. 시스템에서는 2.1에서 7개의 분류로 태그한 문자메시지 중 5,102개의 문자메시지를 n-gram으로 분석하고, 각 n-gram의 tf와 df를 구하여 유사도 계산에 사용한다. tf는 문자메시지 내에서의 문자열의 빈도, df는 5,102개 전체 문자메시지 중에서 문자열을 포함한 메시지의 개수이다.

아래 수식 (1)은 문자메시지간 유사도를 계산한다. 식에서 s는 신규 문자메시지, c는 기분류된 문자메시지이다. s와 c에서 공통으로 발생하는 n-gram t_i 의 s와 c에서의 tf합을 t_i의 df로 나누면, s와 c에 의미 있는 공통 단어가 많을수록 큰 값을 얻을 수 있으며, 이는 두 문자메시지간의 유사도로 볼 수 있다. 최종적으로 식 (2)를 통해 신규 문자메시지와 가장 유사한 기분류된 문자메시지의 분류 k로 결정한다.

$$sim(s,c) = \sum_{t_i \in s \cap c} \frac{tf(t_i,s) + tf(t_i,c)}{df(t_i)} \quad (1)$$

$$class(s) = \arg max_k \max_{c \in C_k} sim(s,c) \quad (2)$$

2.2와 2.3의 내용에 기반한 자동 분류는 100%의 정확

도를 보장할 수 없다. 시스템에서는 레이블 DB에 저장된 전화번호별 분류 통계에 기반하여 최종 분류를 결정하여 정확도를 향상시킨다. 레이블 DB에 대해서는 아래에서 설명한다.

3. 어플리케이션 구현 및 동작

그림 4는 분류를 위한 시스템의 동작 순서를 보인다. 어플리케이션을 처음 실행하면 메시지 분류를 위한 권한을 획득한다. 메시지DB로부터 메시지를 읽어 사용자 단말기의 연락처에 등록된 번호로부터 수신된 메시지의 경우 대화 카테고리로 분류를 수행하고, 등록되지 않은 번호로부터 수신된 경우 문자메시지를 서버에 전송하여 카테고리 분류를 수행한다. 서버로부터 분류가 완료된 전화번호와 해당 번호로부터 수신된 문자메시지가 분류된 카테고리 번호를 수신해 레이블 DB를 생성한다.

레이블 DB는 전화번호와 해당 전화번호로부터 수신된 문자의 분류 결과를 저장하는 데이터베이스로, 각 발신번호 T_i의 문자메시지들이 분류 C_k 각각으로 몇 번 분류되었는지를 저장한다. 각 발신번호는 문자메시지의 분류에 1:1 대응될 것으로 예상할 수 있으며, T_i의 문자메시지가 주로 C_j 분류로 결정된다면 이러한 통계는 분류 정확도를 높이는데 기여할 수 있다. 시스템에서는 레이블 DB를 참조해 분류 통계가 가장 높은 분류를 해당 전화번호의 분류로 선택하여, 내용기반 분류의 오류를 보정한다.

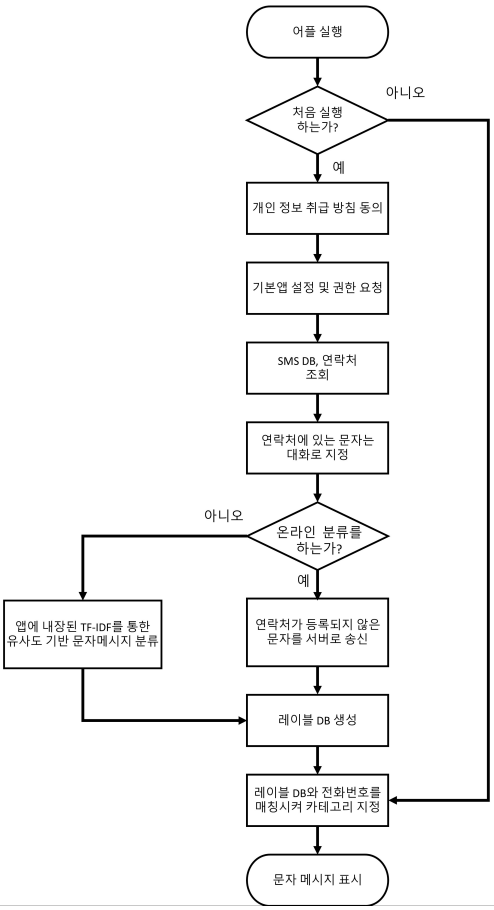

그림 4 어플리케이션 동작 순서도

그림 5는 구축된 어플리케이션의 화면을 보인다. 그림과 같이 전체에서는 수신 순서로 메시지를 나열하여 최신 메시지를 확인할 수 있고, 탭의 형태로 각 분류의 메시지를 보인다. 전체 탭에서는 좌측에 분류를 제시하며 각 탭에서 해당 분류의 문자만 나열한다. 만일 택배 분류의 문자만을 삭제하여 정리하고 싶으면 택배 탭에서 모든 문자를 선택하여 삭제를 수행하면 전체 탭에서도 택배 문자는 제거된다. 공공기관에서 발송된 중요한 문자를 확인하고 싶으면 공공기관 탭에서의 빠르고 편하게 해당 문자를 확인할 수 있어, 사용자의 문자메시지 사용의 편의를 극대화할 수 있다.

4. 실험 및 평가

온라인 기반의 문자메시지 분류에 대한 실험에서는 수동으로 분류한 5,802개의 문자메시지 중 3,942개를 훈련 데이터로, 1,160개를 검증데이터로, 700개를 실험데이터로 사용하였다. 실험데이터에서는 각 분류별 문자메시지

그림 5 문자메시지 분류 후 어플리케이션 화면

의 수를 100개로 동일하게 사용하였다.

표 1은 학습데이터에 기반한 문자메시지 분류에서 각기 다른 차원으로 CBOW와 Skip-gram 각각을 적용한 경우의 분류 정확도와 수행시간을 보인다. CBOW방식의 경우 훈련 속도가 빠르나 빈도수가 낮은 단어에 대해 낮은 정확도를 가진 반면 Skip-gram방식의 경우 훈련 속도가 느리나 빈도수가 낮은 단어에도 좋은 결과를 보인다. CBOW 방식에 비해 Skip-gram의 경우 구문론적인 부분에서는 좋지 못한 결과를 갖지만 의미론적인 부분에서 더 좋은 성능을 보이는 것으로 알려져 있다[7]. 문자메시지의 분류는 구문론적인 부분보다 의미론적인 부분이 더 중요하며, 표 1의 결과에서도 Skip- gram방식이 나은 성능을 나타냄을 확인할 수 있었다. 또한 Skip-gram방식의 경우 차원이 올라갈수록 학습시간의 증가량에 비해 정확도 향상이 미미하여, 단어벡터의 차원으로 300차원 벡터를 사용했다. 문장벡터의 경우 해당 문장에 들어간 단어벡터 값의 평균으로 정했다. 결과에서는 비교적 작은 크기의 학습데이터만으로도 90.96%의 자동 분류 결과를 얻을 수 있었다.

표 1 CBOW와 Skip-gram 방식의 차원별 평가 결과

	차원	정확도(%)	수행시간(초)
CBOW	100	87.06	0.383
	200	87.37	0.478
	300	87.50	0.547
Skip-gram	100	90.52	0.655
	200	90.80	0.734
	300	**90.96**	0.886

표 2는 2.3에서 소개한 오프라인 문자메시지 분류의 정확도를 보인다. 수동으로 분류한 5,802개의 문자메시지 중 5,102개를 기존 분류로 사용하고 각 분류별 100개의 총 700개의 문자메시지에 대한 정확도와 수행시간을 나타낸다. 오프라인 분류의 경우 기분류된 문자메시지에 없는 단어로 구성된 문자메시지는 분류가 불가능하다. 표에서 추출률은 n-gram으로 분류가 된 문자메시지의 비율로, n이 커질수록 동일한 n-gram이 기존 문자에 존재하지 않아 추출률이 낮아지는 결과를 보인다.

표 2에 따르면 N-gram방식을 사용했을 경우 N이 2일 때 가장 높은 정확도를 보여주고 있으나, 수행시간은 가장 느린 것을 알 수 있다. 수행시간은 700개 전체에 대한 결과로 문자메시지 당 0.02초 미만이다. 어절 단위로 분류했을 때는 67.14%라는 낮은 정확도이나 가장 빠른 수행시간을 보여주고 있다.

표 2 N-gram으로 수행한 문장 분류의 정확도 및 수행시간

분류방식	N	추출률(%)	정확도 (%)	수행시간(초)
N-gram	1	100.00	65.57	15.52
	2	99.71	92.71	13.21
	3	97.57	91.57	11.34
	4	**94.14**	**90.29**	**10.44**
	5	90.00	90.29	10.45
	6	86.86	84.29	9.80
	7	84.57	82.14	9.05
어절 단위		94.57	67.14	7.17

시스템에서는 서버로의 문자메시지 전송여부를 사용자가 선택할 수 있게 하였으며, 서버전송을 거부한 경우에는 문자메시지 분류의 정확도, 추출률, 수행시간이 적절한 조화를 이룬 4-gram을 적용했다. 결과에서는 추출률 94.14%에 90.29%의 정확도로 문자메시지 분류에서는 85% 가량의 성능으로, 오프라인에서도 실용성 있는 결과를 얻을 수 있었다.

5. 결론

본 논문은 사용자의 문자메시지의 다중 분류를 수행하는 방법에 대해 제시했다. 온라인 분류의 경우 사용자의 개인 정보를 서버로 전송하여 분석하므로 사용자는 이에 대한 반감과 개인 정보 수집에 대한 불신이 있을 수 있다. 이 부분을 해결하기 위해 오프라인 기반 유사도 분류를 구현했다.

향후 연구로는 모바일 단말에서의 기계학습을 구현해 사용자 개개인에 맞는 맞춤형 문자메시지 분류를 진행하고 있다.

참고문헌

[1] "문자메시지 사용 실태 조사-두잇서베이 설문조사." 두잇서베이. 2016년 10월 13일 수정, 2019년 09월 03일 접속, http://www.dooit.co.kr/survey/report/index/180658/all.

[2] "SKT, 문자메시지 앱 '여름' 서비스 종료-아시아경제." 아시아경제. 2017년 05월 17일 수정, 2019년 09월 03일 접속, https://www.asiae.co.kr/article/2017051710463929548.

[3] 손대능, 이정태, 이승욱, 신중휘, 임해창. “문자메시지의 특성을 고려한 한국어 모바일 스팸필터링 시스템.” 한국산학기술학회논문지 11권, no.7, 2595-2602. 2010.

[4] 이현영, 강승식. "워드 임베딩과 딥러닝 기법을 이용한 SMS 문자메시지 필터링 ." 스마트미디어저널, 7권, no.4, 13-18, 2018. http://www.riss.kr/link?id=A105982597.

[5] "형태소 분석 및 품사 태깅", Konlpy. n.d. 수정, 2019년 09월 04일 접속, https://konlpy-ko.readthedocs.io/ko/v0.4.3/morph/#comparison-between-pos-tagging-classes.

[6] “딥러닝 학습 기술들”, ratsgo’s blog, 2017년 04월 22일 수정, 2019년 09월 04일 접속, https://ratsgo.github.io/deep%20learning/2017/04/22NNtricks/.

[7] Mikolov, T., Chen, K., Corrado, G., and Dean, J. Efficient estimation of word representations in vector space. 2013. arXiv:1301.3781v3.