International Journal of Latest Engineering and Management Research (IJLEMR)
ISSN: 2455-4847
www.ijlemr.com Volume 1 Issue 3 ǁ April 2016 ǁ PP.12-15

K-Best Compression

Nishant Shekhar[1], ShantanuSandilya[2],Vipul Raj Singh[3],
Sonam Kumari[4]
[1234]*(IT,IIMT College Of Eng.,India)*

ABSTRACT -A well-known way to deal with sentence compressionis to figure the errand as a compelled streamlining issue and settle it with whole number direct programming (ILP) apparatuses. Tragically, reliance on ILP may make the compressor restrictively moderate, and in this way guess systems have been proposed which are regularly unpredictable and offer a moderate increase in velocity. As an option arrangement, we present a novel pressure calculation which produces k-best compressions depending on neighbourhood erasure choices. Our calculation is two requests of extent quicker than a late ILPbased technique while delivering better compressions.Moreover, a broad assessment shows that the nature of compressions does not debase much as we move from single best to main five results.

KEYWORDS -ILP,Apparatuses,Compressions,Streamlining, k-best.

1. INTRODUCTION

There has been a surge in sentence pressure research in the previous decade in view of the guarantee it holds for extractive content rundown and the utility it has in the time of cell phones with little screens. Like content rundown, extractive approaches which don't bring new words into the outcome have been especially prevalent. There, the fundamental test lies in knowing which words can be erased without contrarily influencing the data substance or grammaticality of the sentence. Given the many-sided quality of the pressure assignment (the number of conceivable yields is exponential), numerous frameworks outline it, here and there consolidated with rundown, as an ILP issue which is then tackled with off-the-rack devices (Martins and Smith, 2009; Berg-Kirkpatrick et al., 2011; Thadani and McKeown, 2013). While ILP plans are clear and the interpretation to an ILP issue is frequently normal (Clarke and Lapata, 2008), they accompany a high arrangement fetched and restrictively long preparing times (Woodsendand Lapata, 2012; Almeida and Martins, 2013). In this manner, hearty calculations equipped for producing useful what's more, linguistically right compressions at much speedier running times are still attractive. Towards this objective, we propose a novel administered sentence pressure strategy which joins nearby cancellation choices with a recursive methodology of getting most likely compressions at each hub in the tree. To create the top-scoring pressure a solitary tree traversal is required. To amplify the k-best rundown with a k + 1th pressure, the calculation needs O(m n) examinations where n is the hub check and m is the normal spreading component in the tree. Imperatively, estimated look methods like shaft pursuit (Galanis and Androutsopoulos, 2010; Wang et al., 2013), are not required. Contrasted and a late ILP technique (Filippova and Altun, 2013), our calculation is two requests of extent quicker while delivering shorter compressions of meet quality. Both techniques are directed and utilizethe same preparing information and components. The outcomes showthat great comprehensibility and education, asseen by human raters, can be accomplished withoutweakening calculation proficiency. Moreover, both scores stay high as one moves from the top result to the main five. As far as anyone is concerned we are the first toreport assessment comes about past single best yield. We show an augmentation to the calculation where we tradeoff the insurance of acquiring the top scoring answer for the advantage of scoring a hub subset in general. This augmentation just modestly influences the running time while disposing of a source of imperfect compressions.

2. THE TOP-DOWN APPROACH

Our approach is syntax-driven and operates on dependencytreesThe input tree is prunedto obtain a valid subtree from which a compression is read off. The pruning decisions are carried out based on predictions of a maximum entropy classifier which is trained on a parallel corpora with a rich feature set (Sec. 2.2). Section 2.3 explains how to

2.1 PREPROCESSING

Similar to previous work, we have a special treatmentfor function words like determiners, prepositions, auxiliary verbs. Unsurprisingly, dealing with function words is much easier than deciding whether a content word can be removed. Approaches which use a constituency parser and prune edges pointing to constituents, deal with function words implicitly (Berg-Kirkpatrick et al., 2011; Wang et al., 2013).

Approaches which use a dependency representation either formulate hard constraints (Almeida & Martins,2013), or collapse function words with their heads. We use the latter approach and transform every input tree (Nivre, 2006) following Filippova&Strube (2008) and also add edges from the dummy root to finite verbs. Finally, we run an entity tagger and collapse nodes referring to entities.

2.2 ESTIMATING DELETION PROPERTIES

The supervised component of our system is a binary maximum entropy classifier (Berger et al., 1996) which is trained to estimate the probability of deleting an edge given its local context. In what follows, we are going to refer to two probabilities, pde(e).

$$p_{del}(e_{n;m}) + p_{ret}(e_{n;m}) = 1; \quad (1)$$

where del stands for deleting, ret stands for retaining edge e from node n to node m, and pdel(en;m) is estimated with MaxEnt.The features we use are inspired by most recent work (Almeida & Martins, 2013; Filippova&Altun, 2013; Wang et al., 2013) and are as follows: syntactic: edge labels for the child and its siblings; NE type and PoS tags; lexical: head and child lemmas; negation; concatenation of parent lemmas and labels; numeric: depth; node length in words and characters; children count for the parent and the child. Note that no feature refers to the compression generated so far and therefore the probability of removing an edge needs to be calculated only once on a first tree traversal.

2.3 OBTAINING TOP SCORING COMPRESSION

To find the best compression of the sentence we startat the dummy root node and select a child n withthe highest pret(eroot;n). The root of the exampletree in Figure 1 has three children (said2, robbed6,was arrested12). Assuming that pret's for the threepredicates are :07; :5; :9, the third child is selected.Since pdel and pret sum to one, this implies that every edge with pret< 0:5 is deleted. However, we can take any _ 2 [0; 1] to be a threshold for deciding between keeping vs. deleting an edge and linearly scalepdel and pret so that after scaling ^pdel< 0:5 if and only if pdel.

3. ADDING A NODE SUBSET SCORER

On the first pass, the top-down compressor attemptsto find the best possible children subset of every node by considering every child separately and making the retain-or-delete decisions independently of one another. How conservative or aggressive the algorithm is, is determined by a single parameter [0; 1] which places a boundary between the two decisions. With smaller values of _ a low probability of deletion (pdel) would suffice for a node child to be removed. Conversely, a greater value of _ would mean that only children about which the classifier is fairly certain that they must be deleted would be removed. Unsurprisingly, the value of _ is hard to optimize as it may be too low or too high, depending on a node. While retaining a child which could be dropped would not result in an ungrammatical sentence, omitting an important argument may make the compression incomprehensible. When doing an error analysis on a development set, we did not encounter many cases where the compression was clearly ungrammatical due to a wrongly omitted argument. However, results like that do have a high cost and thus need to be addressed.

4. EVALUATION

The purpose of the evaluation is to validate the followingtwo hypotheses, when comparing the new algorithm with a competitive ILP-based sentencecompressor (Filippova&Altun, 2013):

1. The top-down algorithm was designed to performlocal decisions at each node in the parsetree, as compared to the global optimizationcarried out by the ILP-based compressor. We want to verify whether the local model can attainsimilar accuracy levels or even outperformthe global model, and do so not only for thesingle best but the top k results.

2. Automatic ILP optimization can be quite slowwhen the number of candidates that need to be evaluated for any given input is large. We wantto quantify the speed-up that can be attainedwithout a loss in accuracy by taking simpler,local decisions in the input parse tree.

5. EVALUATION SETTINGS

Training, development and test set The aligned sentences and compressions were collected using the procedure described in Filippova&Altun(2013). The training set comprises 1,800,000 items, each item consisting of two elements: the first sentence in a news article and an extractive compression obtained by matching content words from the sentence with those from the headline (see Filippova&Altun (2013) for the technical details). A part of this set was held out for classifiers evaluation and development. For testing, we use the dataset released by Filippova&Altun (2013)1. This test set contains 10,000 items, each of which includes the original sentence and the extractive compression and the URL of the source document. From this set, we used the first 1,000 items only, leaving the remaining 9,000 items unseen, reserved for possible future experiments. We made sure that our training set does not include any of the sentences from the test set. The training set provided us with roughly 16 million edges for training MaxEnt with 40% of positive examples (deleted edges). For training the perceptron classifier we had about 6 million nodes at our disposal with the instances distributed over the five classes as follows:

0 1 2 3 4+

Baseline We used the recent ILP-based algorithm of Filippova&Altun (2013) as a baseline. We trained the compressor with all the same features as our model (Sec. 2.2) on the same training data using an averaged perceptron (Collins, 2002). To make this system comparable to ours, when training the model, we did not provide the ILP decoder with the oracle compression length so that the model learned to produce compressions in the absense of length argument.

6.AUTOMATIC EVALUATION

To measure the quality of the two classifiers (Max-Ent from Sec. 2.2 and perceptron from Sec. 3), we performed a first, direct evaluation of each of them on a small held out portion of the training set. The MaxEnt classifier predicts the probability of deleting an edge and outputs a score between zero and one. Figure 3 plots precision, recall and F1-score at different threshold values. The highest F1-score is obtained at 0.45. Regarding the perceptron classifier that predicts the number of children that we should retain for each node, its accuracy and per-class precision and recall values are given in Table 2. Acc 0 1 2 3 4+ 72.7 69 / 63 75 / 78 76 / 81 60 / 42 44 /16It is important to point out the difference in compressionrates between ILP and TOP-DOWN: 47%vs. 38% (the average compression rate on the testset is 40.5%). Despite a significant advantage due tocompression rate (Napoles et al., 2011, see next subsection),ILP performs slightly worse than the proposedmethods.Finally, 0
As can be seen, in both cases there is a sharp drop between the top two compressions but further scores are very close. Since the test set only contains a single oracle compression for every sentence, to understand how
big the gap in quality really is, we need an evaluation with human raters.

7.MANUAL EVALUATION

The first 100 items in the test data were manually rated by humans. We asked raters to rate both readability and informativeness of the compressions for the golden output, the baseline and our systems. For both metrics a 5-point Likert scale was used, and three ratings were collected for every item. Note that in a human evaluation between ILP and TOP-DOWN(+ NSS) the baseline has an advantage because (1) it prunes less aggressively and thus has more chances of producing a grammaticaly correct and informative outputs, and (2) it gets a hint to the optimal compression length in edges. We have used Intra- Class Correlation (ICC) (Shrout& Fleiss, 1979;The evaluation template and rated sentences are included in the supplementary material. Cicchetti, 1994) as a measure of inter-judge agreement. ICC for readability was 0.59 (95% confidence interval [0.56, 0.62]) and for informativeness it was
0.51 (95% confidence interval [0.48, 0.54]), indicating fair reliability in both cases.
Results are shown in Tables 5 and 6. As in the automatic evaluations, the two Top-down systems produced indistinguishable results, but both are significantly better than the ILP baseline at 95% confidence. The top-down results are also now indistinguishable
from the extractive compressions.

8. EFFICIENCY

The average per-sentence processing time is 32,074 microseconds (Intel Xeon machine with 2.67 GHz CPU) using ILP, 929 using TOP-DOWN + NSS, and using TOP-DOWN. This means that we have obtainedalmost a 50x performance increase over ILP.The1,000 sentences in the test set with sentence length measured in tokens. For obtaining k-best solutions, the decrease in time is even more remarkable: the average time for generating each of the top-5 compressions using ILP is 42,213 microseconds, greater than that of the single best result. Conversely, the average time for each of the top-5 results decreases to 143 microsec

onds using TOP-DOWN, and 195 microseconds using TOP-DOWN + NSS, which means a 300x improvement. The reason is that the Top-down methods,in order to produce the top-ranked compression, have already computed all the per-edge predictions (and the per-node NSS predictions in the case of TOP-DOWN + NSS), and generating the next best solutions is cheap.

9.CONCLUSION

We presented a fast and accurate supervised algorithm for generating k-best compressions of a sentence. Compared with a competitive ILP-based system,our method is 50x faster in generating the bestresult and 300x faster for subsequent k-best compressions.Quality-wise it is better both in terms ofreadability and informativeness. Moreover, an evaluationwithhumanraters demonstrates that the qualityof the output remains high for the top-5 results.

REFERCENCES

[1]. Almeida, M. B. & A. F. T. Martins (2013). Fast and\robust compressivesummarization with dual decompositionand multi-task learning. In Proc.OfACL-13.Berg-Kirkpatrick, T., D. Gillick& D. Klein (2011).

[2]. Jointly learning to extract and compress. In Proc.of ACL-11.Berger, A., S. A. Della Pietra& V. J. Della Pietra(1996). A maximum entropy approach to naturallanguage processing. Computational Linguistics,22(1):39–71.

[3]. Cicchetti, D. V. (1994). Guidelines, criteria, andrules of thumb for evaluating normed and standardizedassessment instruments in psychology.Psychological Assessment, 6(4):284.

[4]. Clarke, J. & M. Lapata (2006). Constraint-based sentence compression: An integer programmingapproach. In Proc. of COLING-ACL-06 PosterSession, pp. 144–151.

[5]. Clarke, J. & M. Lapata (2008). Global inference for sentence compression: An integer linear programming approach. Journal of Artificial IntelligenceResearch, 31:399–429.

[6]. Collins, M. (2002).Discriminative training methodsfor Hidden Markov Models: Theory and experiments with perceptron algorithms. In Proc.Of EMNLP-02, pp. 1–8.Filippova, K. & Y. Altun (2013).

[7]. Overcoming thelack of parallel data in sentence compression.InProc. of EMNLP-13, pp. 1481–1491.

[8]. Filippova, K. & M. Strube (2008). Dependency tree based sentence compression. In Proc. of INLG- 08, pp. 25–32.

[9]. Fillmore, C. J., C. R. Johnson & M. R. Petruck (2003). Background to FrameNet.International Journal of Lexicography, 16:235–260.

[10]. Freund, Y. & R. E. Shapire (1999). Large margin classification using the perceptron algorithm. MachineLearning, 37:277–296. Galanis, D. & I. Androutsopoulos (2010).

[11]. An extractive supervised two-stage method for sentence compression. In Proc. of NAACL-HLT-10, pp.885–893.

[12]. Galley, M. & K. R. McKeown (2007).Lexicalized Markov grammars for sentence compression.In Proc. of NAACL-HLT-07, pp. 180–187.

[13]. Grefenstette, G. (1998). Producing intelligent telegraphic text reduction to provide an audio scanning service for the blind. In Working Notes ofthe Workshop on Intelligent Text Summarization,Palo Alto, Cal., 23 March 1998, pp. 111–117.Huang, L. & D. Chiang (2005). Better k-best parsing.

[14]. Technical Report MS-CIS-05-08: University of Pennsylvania. Jing, H. & K. McKeown (2000). Cut and paste based text summarization. In Proc. of NAACL-00, pp. 178–185.

[15]. Knight, K. & D.Marcu (2000). Statistics-based summarization – step one: Sentence compression. In Proc. of AAAI-00, pp. 703–711.

[16]. Martins, A. F. T. & N. A. Smith (2009).Summarization with a joing model for sentence extraction and compression.In ILP for NLP-09, pp. 1–9.

[17]. McDonald, R. (2006). Discriminative sentence compression with soft syntactic evidence.In Proc.OfEACL-06, pp. 297–304.

[18]. Napoles, C., C. Callison-Burch & B. Van Durme(2011). Evaluating sentence compression: Pitfalls and suggested remedies. In Proceedings ofthe Workshop on Monolingual Text-to-text Generation,Prtland, OR, June 24 2011, pp. 91–97.Nivre, J. (2006).

[19]. Inductive Dependency Parsing. Springer.Nomoto, T. (2009).A comparison of model free versus model intensive approaches to sentence compression.In Proc. of EMNLP-09, pp. 391–399.

[20]. Qian, X. & Y. Liu (2013). Fast joint compression and summarization via graph cuts.In Proc.OfEMNLP-13, pp. 1492–1502.

[21]. Riezler, S., T. H. King, R. Crouch & A. Zaenen (2003).Statistical sentence condensation usingambiguity packing and stochastic disambiguationmethods for Lexical-Functional Grammar.In Proc. of HLT-NAACL-03, pp. 118–125.Shrout, P. E. & J. L. Fleiss (1979).