Meteor++ 2.0: Adopt Syntactic Level Paraphrase Knowledge into Machine Translation Evaluation

Yinuo Guo, Junfeng Hu[*]
Key Laboratory of Computational Linguistics, School of EECS, Peking University
{gyn0806, hujf}@pku.edu.cn

Abstract

This paper describes Meteor++ 2.0, our submission to the WMT19 Metric Shared Task. The well known Meteor metric improves machine translation evaluation by introducing paraphrase knowledge. However, it only focuses on the lexical level and utilizes consecutive n-grams paraphrases. In this work, we take into consideration syntactic level paraphrase knowledge, which sometimes may be skip-grams. We describe how such knowledge can be extracted from Paraphrase Database (PPDB) and integrated into Meteor-based metrics. Experiments on WMT15 and WMT17 evaluation datasets show that the newly proposed metric outperforms all previous versions of Meteor.

1 Introduction

Accurate evaluation of machine translation (MT) plays an important role in measuring improvement in system performance. Since human evaluation is time-consuming and expensive, automatic metrics for MT have received significant attention in the past few years. A lot of MT evaluation metrics from different perspective have been proposed to measure how close machine-generated translations are to professional human translations such as BLEU (Papineni et al., 2002), Meteor (Banerjee and Lavie, 2005), TER (Snover et al., 2006) etc. Because Meteor has the ability to employ various linguistic language features and resources easily, a lot of improved versions has been put forward continuously (Lavie and Denkowski, 2009; Denkowski and Lavie, 2010a,b, 2011, 2014; Guo et al., 2018). The Meteor-Next (Denkowski and Lavie, 2010a,b) extends the Meteor to phrase-level with the support of paraphrase tables. It's clear that this knowledge incorporated into matching procedure do help the metric reach a higher correlation with the human scores.

In previous work, phrases in paraphrase table are defaulted to be consecutive n-grams which mainly draw on the lexical level. What's more, skip n-gram (Huang et al., 1993) paraphrases whose components need not be consecutive also capture many meaning-preserving syntactic transformations. The original Meteor-based metrics only pay attention to consecutive string matching, they perform badly when reference-hypothesis pairs contain skip n-grams. Using the pair **(protect...from, protect...against)** for an example, the two different prepositions **from** and **against** will bring a miss-matching and then have a negative effect on the Meteor score. Obviously, these two words are equivalent when appearing simultaneously with the verb **protect**. What's more, **from** and **against** here mainly support the sentence structure and contribute little on semantic expression.

In this paper, we seek to directly address the problem mentioned before by adopting a syntactic-level language resource into Meteor. Taking advantage of the large Paraphrase Database (Ganitkevitch et al., 2013; Pavlick et al., 2015), we automatically extract a subset of syntax PPDB which contains skip n-grams. To demonstrate the efficacy of this knowledge, we raise an improved version of the Meteor incorporated with that via an extra parallel syntax stage. Our extended metric, Meteor++ 2.0, shows an improvement in the correlation with the human scores on most of the language pairs.

We organize the remainder of the paper as follows: Section 2 describes the traditional Meteor scoring. Section 3 presents the syntactic level paraphrase table acquisition and model details. Section 4 is devoted to the experiments and results. The conclusions follow in the final section.

2 Traditional Meteor Scoring

The Meteor metric based on a general concept of flexible unigram matching, unigram precision and unigram recall, including the match of words that are simple morphological variants of each other by the identical stem and words that are synonyms of each other. For a single hypothesis-reference pair, the space of possible alignments is constructed by exhaustively identifying all possible matches between the sentences according to the following matchers with different weight.

- *Exact:* Words are matched if and only if their surface forms are identical.
- *Stem:* Words are stemmed using a language appropriate Snowball Stemmer (Porter, 2001) and matched if the stems are identical.
- *Synonym:* Words are matched if they are both members of a synonym set according to the WordNet (Miller, 1998) database.
- *Paraphrase:* Phrases are matched if they are listed as paraphrases in a paraphrase table.

Alignment resolution is conducted as a beam search using a heuristic based on the specified criteria.The final alignment is then resolved as the largest subset of all matches to meet the following criteria in order of importance:

1. Require each word in each sentence to be covered by zero or one match.
2. Maximize the number of covered words across both sentences.
3. Minimize the number of chunks, where a chunk is defined as a series of matches that is contiguous and identically ordered in both sentences.
4. Minimize the sum of absolute distances between match start indices in the two sentences. (Break ties by preferring to align phrases that occur at similar positions in both sentences.)

Once the final alignment is selected, the Meteor calculates weighted precision P and recall R. For each matcher (m_i), it counts the number of content and function words covered by matches of ith type in the hypothesis ($m_i(h_c)$, $m_i(h_f)$) and reference ($m_i(r_c)$, $m_i(r_f)$), $|h_f|$ and $|r_f|$ mean the total number of function words in hypothesis and reference, $|h_c|$ and $|r_c|$ mean the total number of content words in hypothesis and reference.

$$P = \frac{\sum_i w_i \cdot (\delta \cdot m_i(h_c) + (1-\delta) \cdot m_i(h_f))}{\delta \cdot |h_c| + (1-\delta) \cdot |h_f|} \tag{1}$$

$$R = \frac{\sum_i w_i \cdot (\delta \cdot m_i(r_c) + (1-\delta) \cdot m_i(r_f))}{\delta \cdot |r_c| + (1-\delta) \cdot |r_f|} \tag{2}$$

The parameterized harmonic mean of precision P and recall R then calculated:

$$F_{mean} = \frac{P \cdot R}{\alpha \cdot P + (1-\alpha) \cdot R} \tag{3}$$

To account for gaps and differences in word order, a fragmentation penalty is calculated using the total number of matched words (m, averaged over hypothesis and reference) and number of chunks(ch):

$$Pen = \gamma \cdot (\frac{ch}{m})^{\beta} \tag{4}$$

The Meteor score is then calculated:

$$Score = (1 - Pen) \cdot F_{mean} \tag{5}$$

The parameters α, β, γ, δ and $w_i ... w_n$ are tuned to maximize correlation with human judgments.

3 Our Approach

In this section, we firstly present the syntactic level paraphrase table acquisition in 3.1 and then we introduce how to integrate this knowledge resource into Meteor in 3.2.

Element A	Element B
assist _ in	help _ to
protect _ from	protect _ against
the turkish _	the _ of turkey
feel _ is	believe that _ is
_ administration	of _ management
give	provide _ with
ask _ to do	ask that _
depressing _ of	depressing _ from
issue	the _ number of

Table 1: Some examples of our extracted Syntactic Level Paraphrase Table. Note that '_' is the placeholder which can be skipped over.

3.1 Syntactic Level Paraphrase Table Acquisition

Syntactic paraphrases always capture meaning-preserving syntactic transformations but gain less attention than lexical paraphrases in Meteor-based metrics. In this work, we benefit from the widely

Reference	We(0) will(1) get(2) the(3) boys(4) ready(5) to(6) go(7) again(8) said(9) donnelly(10)
Hypothesis	We(0) will(1) prepare(2) the(3) boy(4) back(5) to(6) action(7) donnelly(8) promises(9)

Index	Reference	Hypothesis	Match Type	Match Weight
0	we(0)	we(0)	*Exact*	[1.0]
1	will(1)	will(1)	*Exact*	[1.0]
2	get(2) _ _ ready(5)	prepare(2)	*Syntax*	[0.4, 0, 0, 0.8]
3	the(3)	the(3)	*Exact*	[1.0]
4	boys(4)	boy(4)	*Synonym*	[0.8]
5	-	-	-	-
6	to(6)	to(6)	*Exact*	[1.0]
7	go(7)	-	-	-
8	again(8)	-	-	-
9	said(9)	-	-	-
10	donnelly(10)	donnelly(8)	*Exact*	[1.0]

Table 2: An example of alignment result between the reference-hypothesis pair.The weights of *Exact*, *Stem*, *Synonym*, *Paraphrase* and *Syntax* are set to be [1.0, 0.6, 0.8, 0.6, 0.4]

used paraphrase resource PPDB2.0 (Pavlick et al., 2015) and try to bring syntactic level knowledge into Meteor evaluation. Here we mainly focus on skip n-gram paraphrases whose components are not consecutive in appearance such as examples shown in Table 1. Note that here we filter those pairs in which both elements are consecutive n-grams because they will be duplicate with the pairs in existing paraphrase table of Meteor.

The PPDB divides the database into six sizes for several languages according to three perspectives, from *S* to *XXXL* on the lexical, phrasal and syntactic level. We build our own syntactic level paraphrase table using the *XXXL* syntax PPDB:Eng which contains over 140 million syntax paraphrase pairs. Then we use regular expressions to extract the skip n-gram paraphrases with the following criteria and hold out about 27 million pairs. The paraphrase pair in the following descriptions means two phrases which are listed as paraphrases in our syntactic level paraphrase shaped like (Element A, Element B) in Table 1.

- Each phrase in one paraphrase pair should be not consecutive in appearance .
- Each phrase in one paraphrase pair should contain at least one content word.
- The length difference between the two phrases in one paraphrase pair should less than the threshold.

Compared to the prior paraphrase tables, we list two principal differences between them:

- In appearance, we mainly focus on skip n-grams whose components need not be consecutive in the text. In our table, at least one element in each paraphrase pair should have a break by the placeholder '_' which means the position can be any word.
- Lots of pairs have **duplicate** words between the two elements in ours. For the reason that some meaning-preserving syntactic transformations just substitute function words and still have the same content words.

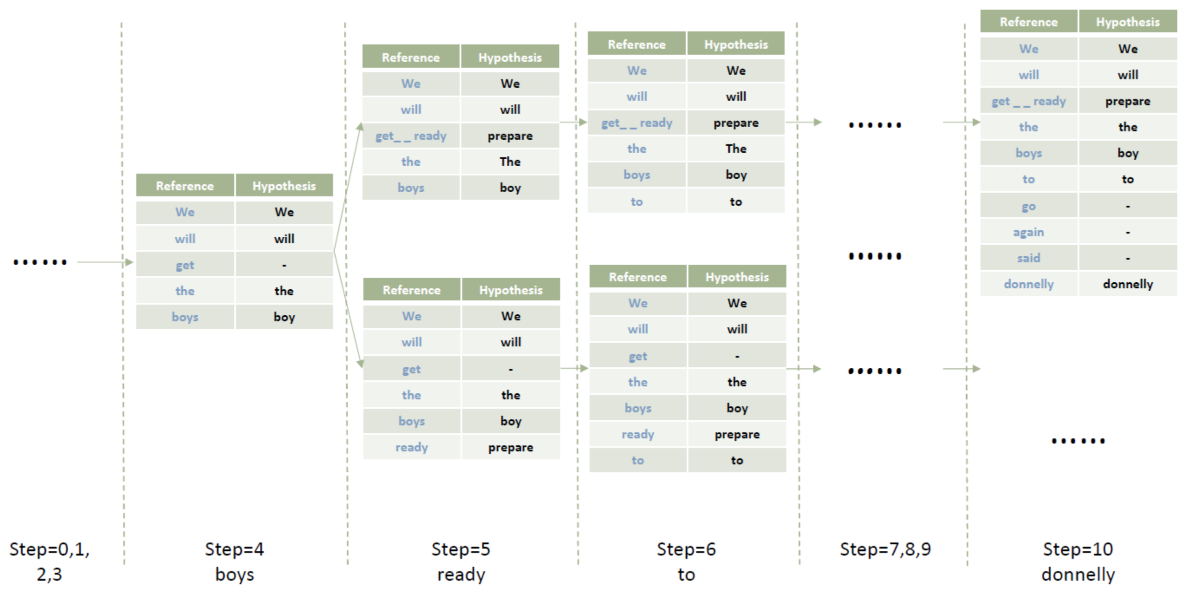
Therefore, treated this knowledge the same way with the previous is unreasonable, and we will discuss the details on how to leverage this language resources under the Meteor framework in the next section.

3.2 Meteor++ 2.0

Under the Meteor framework, Meteor++ 2.0 adds a parallel *Syntax* stage for possible syntactic level paraphrases matching. Due to its difference mentioned in 3.1, we discuss the following modified steps during the matching process.

3.2.1 Possible Alignment Construction

In the extended *Syntax* stage, phrases are matched if they are listed as a pair in our syntactic para-



Reference	Hypothesis
We	We
will	will
get	-
the	the
boys	boy

Reference	Hypothesis
We	We
will	will
get _ _ ready	prepare
the	The
boys	boy

Reference	Hypothesis
We	We
will	will
get	-
the	the
boys	boy
ready	prepare

Reference	Hypothesis
We	We
will	will
get _ _ ready	prepare
the	The
boys	boy
to	to

Reference	Hypothesis
We	We
will	will
get	-
the	the
boys	boy
ready	prepare
to	to

Reference	Hypothesis
We	We
will	will
get _ _ ready	prepare
the	the
boys	boy
to	to
go	-
again	-
said	-
donnelly	donnelly

Figure 1: An example of the modified beam search.

phrase table (3.1). The position of the place-holder '_' can be any word which will be skipped over in this phrase matching. And we only keep those matching pairs with the absolute distances between match start indices in the reference and hypothesis less than the threshold.

In prior paraphrase stage, all words in both Element A and B are set with the same value. While in the *Syntax* stage, we set the different word with a different value in one element of the paraphrase pair. More generally, we set 1.0 as the weight of *Exact* stage and 0.4 as the weight of *Syntax* stage. Consider about the paraphrase pair (**protect _ from, protect _ against**), we suppose the two elements appear in the reference and hypothesis separately. If we assign the weight 0.4 for all the words in this paraphrase pair, there will be a bias with the other *Exact* matching pairs. Because the two **protect** would be an exact matching with the weight 1.0 if *Syntax* stage doesn't exist. In a word, as for the weight assignment in *Sytax* stage , we set the word exact weight if it appears in both elements, and set the word synonym weight if the other element includes its synonym and so on.

Table 2 shows a matching example in a reference-hypothesis pair. The weights of *Exact, Stem, Synonym, Paraphrase* and *Syntax* are set to be [1.0, 0.6, 0.8, 0.6, 0.4]. The (**get _ ready, prepare**) pair is matched in *Syntax* stage, the weight for words **ready** and **prepare** is 0.8, for the reason that they are synonym each other, in other word, it would be matched in an *Synonym* stage if no *Syntax* stage here. And for the word **get**, only matched in the syntax stage, set with the *Syntax* weight 0.4.

3.2.2 Incorporate Syntactic knowledge into Beam Search

The incorporation of the syntactic level paraphrases will bring much more possible matches, therefore it requires a larger beam size which leads a low efficiency. Consider the trade-off between performance and efficiency, we add the syntactic matching pair into the current path until the last word appears during the beam search procedure. Then we look backward to check the state of the other words in this pair, if they are all free to match, we add it into our path.

Figure 1 shows an example in the modified beam search process. At step 4, **get** is an unmatched word in the reference. When comes to the word **ready** in next step, **(get _ ready, prepare)** is a syntax matching pair between the reference-hypothesis. Then we look backward and find that **get** hasn't been matched by others words before, finally, we add the paths with or without **(get _ ready, prepare)** into the current path queue.

4 Experiments

4.1 Setups

To evaluate the impact of our syntactic level paraphrase knowledge, We carry out experiments to compare the performance of Meteor++ 2.0 and

	lang-pair	de-en	fi-en	ru-en	cs-en	tr-en	lv-en	zh-en
WMT2015	Meteor	.615	**.638**	.629	.595	-	-	-
	Meteor++ 2.0 (syntax)	**.621**	.633	**.631**	**.606**	-	-	-
WMT2017	Meteor	.532	.719	**.621**	.555	**.628**	.555	.639
	Meteor++ 2.0 (syntax)	**.535**	**.722**	**.621**	**.561**	**.628**	**.556**	**.646**

Table 3: Comparison of segment-level Pearson correlation between Meteor and Meteor++ 2.0 (syntax) on WMT15 and WMT17 evaluation datasets. The weight of *Syntax* stage in Meteor++ 2.0 is set to be 0.4, other parameters are consistent with the Meteor Universal.

	lang-pair	de-en	fi-en	ru-en	cs-en	tr-en	lv-en	zh-en
WMT2015	Meteor++ (copy)	.630	**.652**	.625	.595	-	-	–
	Meteor++ 2.0 (copy + syntax)	**.634**	.647	**.628**	**.606**	-	-	-
WMT2017	Meteor++ (copy)	.525	.717	.625	.557	**.623**	.562	.644
	Meteor++ 2.0 (copy + syntax)	**.527**	**.721**	**.626**	**.563**	.621	**.565**	**.652**

Table 4: Comparison of segment-level Pearson correlation between Meteor++ (copy) and Meteor++ 2.0 (copy + syntax) on WMT15 and WMT17 evaluation datasets. The weight of *Syntax* stage in Meteor++ 2.0 is set to be 0.4, other parameters are consistent with the Meteor Universal.

other prior Meteor-based metrics using the evaluation datasets on WMT15 and WMT17 to-English pairs. And we tune the weight of *Syntax* stage to maximize the Pearson correlation with human scores on all WMT16 to-English datasets, other parameters are consist of the Meteor Universal. Table 5 shows statistics for each language-pair in WMT15-17, each dataset contains the source sentence, MT output, reference, and human score. And we calculate the Pearson correlation between metric scores and human scores for each language pair.

lang-pair	WMT15	WMT16	WMT17
de-en	500	560	561
fi-en	500	560	561
ru-en	500	560	561
ro-en	-	560	-
cs-en	500	560	561
tr-en	-	560	561
lv-en	-	-	561
zh-en	-	-	561

Table 5: Number of sentences for each language pairs in WMT15-17 evaluation sets.

4.2 Results

Table 3-4 show the Pearson correlation with direct assessment (DA) (Graham et al., 2013) on WMT15 and WMT17 evaluation sets at segment-level. Meteor++ is the previous work in WMT2018 (Guo et al., 2018) integrated with copy knowledge, i.e. words that are likely to be preserved across all paraphrases of a sentence in a given language. Meteor++ 2.0 is the newly proposed one in this paper. In Table 3, we give the comparison between Meteor and Meteor++ 2.0, and Table 4 gives the comparison between Meteor++ and Meteor++ 2.0. **For both prior Meteor-based metrics, the incorporation of the syntactic paraphrase table has a positive influence on almost every to-English language pairs.** Apparently, Meteor++ 2.0 (copy + syntax), the combination with Guo et al. (2018) achieve the best performance in almost every language pair. Hence, we submit Meteor++ 2.0 (copy + syntax) to WMT19 Metric task to-English language pairs.

5 Conclusion

In this paper, we describe the submission of our proposed metric Meteor++ 2.0 for WMT19 Metrics task. Firstly, we extract a syntactic level paraphrase table from the syntax PPDB and list the principle differences between the two paraphrase tables. Secondly, we propose Meteor++ 2.0 incorporated with this language resource. Finally, our metric outperforms all prior Meteor-based metrics on almost every WMT15 and WMT17 to-English language pairs.

6 Future Work

According to the observation of the phrase matches contributed by syntactic level paraphrases, though we benefit a lot from this knowledge resource, some noises are brought at the meantime.

Firstly, in the perspective of the knowledge quality, despite filtering techniques, there are still some unusual, inaccurate or highly context-dependent paraphrases. High-frequency usage always indicates high confidence, so hope our metric can play a role of a quality estimator for paraphrase tables in the future.

Secondly, since the syntax-level knowledge pay more attention on sentence structure, mismatch can not always be avoid. With the help of syntactic tools such as parsing may help take better usage of this knowledge.

7 Acknowledgments

This paper is supported by the National Natural Science Foundation of China (Grant No. 61472017)

References

Satanjeev Banerjee and Alon Lavie. 2005. Meteor: An automatic metric for mt evaluation with improved correlation with human judgments. In *Proceedings of the acl workshop on intrinsic and extrinsic evaluation measures for machine translation and/or summarization*, pages 65–72.

Michael Denkowski and Alon Lavie. 2010a. Extending the meteor machine translation evaluation metric to the phrase level. In *Human Language Technologies: The 2010 Annual Conference of the North American Chapter of the Association for Computational Linguistics*, pages 250–253. Association for Computational Linguistics.

Michael Denkowski and Alon Lavie. 2010b. Meteor-next and the meteor paraphrase tables: Improved evaluation support for five target languages. In *Proceedings of the Joint Fifth Workshop on Statistical Machine Translation and MetricsMATR*, pages 339–342.

Michael Denkowski and Alon Lavie. 2011. Meteor 1.3: Automatic metric for reliable optimization and evaluation of machine translation systems. In *Proceedings of the sixth workshop on statistical machine translation*, pages 85–91. Association for Computational Linguistics.

Michael Denkowski and Alon Lavie. 2014. Meteor universal: Language specific translation evaluation for any target language. In *Proceedings of the ninth workshop on statistical machine translation*, pages 376–380.

Juri Ganitkevitch, Benjamin Van Durme, and Chris Callison-Burch. 2013. Ppdb: The paraphrase database. In *Proceedings of the 2013 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 758–764.

Yvette Graham, Timothy Baldwin, Alistair Moffat, and Justin Zobel. 2013. Continuous measurement scales in human evaluation of machine translation. In *Proceedings of the 7th Linguistic Annotation Workshop and Interoperability with Discourse*, pages 33–41.

Yinuo Guo, Chong Ruan, and Junfeng Hu. 2018. Meteor++: Incorporating copy knowledge into machine translation evaluation. In *Proceedings of the Third Conference on Machine Translation: Shared Task Papers*, pages 740–745.

Xuedong Huang, Fileno Alleva, Hsiao-Wuen Hon, Mei-Yuh Hwang, Kai-Fu Lee, and Ronald Rosenfeld. 1993. The sphinx-ii speech recognition system: an overview. *Computer Speech & Language*, 7(2):137–148.

Alon Lavie and Michael J Denkowski. 2009. The meteor metric for automatic evaluation of machine translation. *Machine translation*, 23(2-3):105–115.

George Miller. 1998. *WordNet: An electronic lexical database*. MIT press.

Kishore Papineni, Salim Roukos, Todd Ward, and Wei-Jing Zhu. 2002. Bleu: a method for automatic evaluation of machine translation. In *Proceedings of the 40th annual meeting on association for computational linguistics*, pages 311–318. Association for Computational Linguistics.

Ellie Pavlick, Pushpendre Rastogi, Juri Ganitkevitch, Benjamin Van Durme, and Chris Callison-Burch. 2015. Ppdb 2.0: Better paraphrase ranking, fine-grained entailment relations, word embeddings, and style classification. In *Proceedings of the 53rd Annual Meeting of the Association for Computational Linguistics and the 7th International Joint Conference on Natural Language Processing (Volume 2: Short Papers)*, volume 2, pages 425–430.

Martin F Porter. 2001. Snowball: A language for stemming algorithms.

Matthew Snover, Bonnie Dorr, Richard Schwartz, Linnea Micciulla, and John Makhoul. 2006. A study of translation edit rate with targeted human annotation. In *Proceedings of association for machine translation in the Americas*, volume 200.