依存構造の連鎖を考慮したニューラル文圧縮

上垣外 英剛　林 克彦　平尾 努　永田 昌明
NTT コミュニケーション科学基礎研究所

1　はじめに

文圧縮とは、原文の大意を保持しつつ、単語を削除することで短い文を生成するタスクである。現在、圧縮文の可読性を担保するために、構文木を刈り込む手法 [6, 9, 1, 3] や、Sequence-to-Sequence (Seq2Seq) を用いる手法 [2] が提案されている。Wang ら [11] はこれらの手法を組み合わせることで最高性能を達成する文圧縮器を実現した。具体的には、LSTM を用いて文中の各単語が圧縮文に採用される重み (確率値) を付与し、重みの和が最大となるよう単語依存構造木を刈り込む。

しかし、Wang らの手法は、依存構造木の葉ノードから順に単語を削除する手法であり、依存構造木の葉に近いノードに重要な単語があったとしても、それが根から離れている、つまり深い位置にある場合には、圧縮文に残すことが難しい。

また、図 1 に示すように、事前調査の結果、Seq2Seq のような依存構造木に頼らない手法であっても、深い依存構造を持つ文では文圧縮の性能が低下することが判明した[*1]。さらに図 2 に示すように、このような深い依存構造木は長文に存在することが多い。長文を短くすることが目的である文圧縮タスクにおいて、深い依存構造木を持つ文の圧縮性能を向上させることは、重要な課題である。

図 3 に深い依存構造を持つ文とその圧縮結果の例を示す。この例は電力輸入協定に関する文であり、国名が重要である。国名を示す句 “from Kyrgyz Republic and Tajikistan” を維持しつつ圧縮を行うには、この句からルートまでに辿る単語 “signed”、“resolution”、“import” も圧縮後の文で維持しなければならない。つまり、依存構造木の深い位置に重要な単語が出現する場合に、それを圧縮文に残すためには依存構造の連鎖を考慮しなければならない。

本稿では、深い依存構造を持つ文の圧縮精度を向上させるために、任意の単語から祖先への依存構造の連鎖を考慮した注意機構を提案し、Seq2Seq に組み込む。

このような連鎖を注意機構で考慮するためには、教師データとして正解となる依存構造木が必要となる。しかし、我々が使用可能な教師データは既存の構文解析器によって解析された結果に限られるため、誤りを含む可能性がある。誤りの影響を避けるため、提案手法は、複数の単語間の依存の可能性を考慮しつつ、単語の削除確率を同時に学習する。

実験の結果、提案手法は Google 文圧縮データセットにおいて F 値で最高精度を達成し、ROUGE-1,2,L スコアにおいても従来手法からの改善が確認された。さらに、評価の対象を深い依存構造を持つ文に限定した場合においても、提案手法による F 値と ROUGE-1,2,L スコアの改善が確認された。

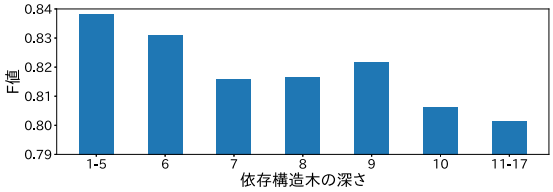

図 1: 依存構造木の深さと Seq2Seq の F 値

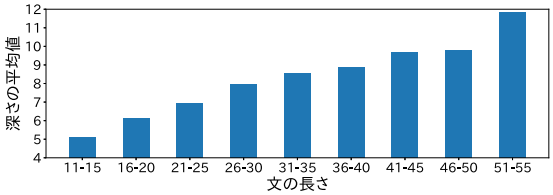

図 2: 文の長さと依存構造木の深さ

2　Seq2Seq による文圧縮

我々が提案する注意機構を組み込む Seq2Seq モデルについて説明する。この Seq2Seq モデルは、Fillipova が提案したモデルの拡張であり、単語埋め込み層、エンコーダ、デコーダ、出力部によって構成されている。

文圧縮は入力単語列 $\mathbf{x} = (x_0, ..., x_n)$ 中の単語 x_t に対し、出力ラベル y_t を (“維持”、“削除”、“文の終了”) から選択するタスクであると考えることができる。Fillipova ら [2] はラベルの予測に Seq2Seq を用いることで、出力ラベル間の依存を考慮した文圧縮を行っている。

単語埋め込み層では、単語 x_t に対する埋め込みベクトル e_t を計算するために、単語の表層、品詞、依存関係ラベルに対する埋め込みベクトル w_i、p_i、r_i を $e_i = [w_i, p_i, r_i]$ のように結合する [11]。なお、[] はベクトルの結合を表す。

エンコーダでは、forward-LSTM と backward-LSTM から構成される両方向 LSTM を用いて、計算された埋め込みベクトル e_t を隠れ状態ベクトル h_t へと変換する。backward-LSTM の最終状態はデコーダの初期状態へと引き継がれる。

デコーダでは、単語埋め込み e_t、y_{t-1} の 3-bit からなる one-hot なベクトル埋め込みと以前の隠れ状態 d_{t-1}(後述) を結合したベクトルを、forward-LSTM を用いて、隠れ状態 $\overrightarrow{s}_t$ へと変換する。

出力部では、ラベル y_t の出力確率を、

$$P(y_t \mid y_{<t}, \mathbf{x}) = softmax(W_o \cdot d_t) \cdot \delta_{y_t}, \quad (1)$$

$$d_t = [h_t, \overrightarrow{s}_t] \quad (2)$$

[*1] F 値の計算には Fillipova らの Seq2Seq [2] を、第 4 節で説明する実験設定に基づいて使用した。

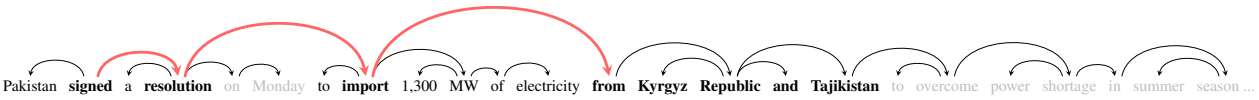

図 3: 深い依存構造木を持つ文とその圧縮結果の例。灰色の単語は削除されたことを示している。

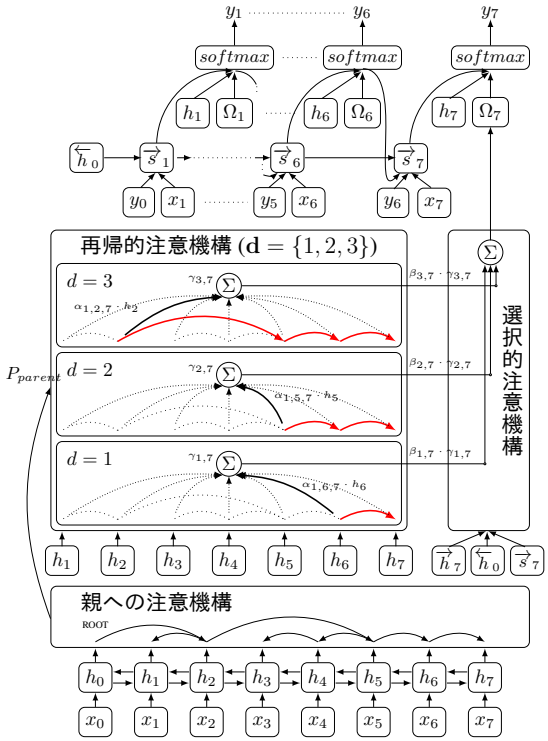

図 4: 提案手法のネットワーク構成図

のように計算する [10]。W_o はソフトマックス層の重み行列を、δ_{y_t} は y_t 番目の要素では 1 を、それ以外では 0 を出力する 2 値ベクトルをそれぞれ表す。

3 提案手法

我々が提案する依存構造の連鎖を考慮した注意機構は、前節にて説明した Seq2Seq の上に構築される。図 4 は提案手法を用いた Seq2Seq の各モジュールとネットワーク間の結合を示している。この図では、入力された単語列に対し、単語 x_7 のラベル y_7 の出力確率を決定する過程が描かれている。y_7 の出力確率は次のように予測される。

1. **親への注意機構**は単語 x_t の親が x_i となる確率 $P_{parent}(x_i|x_t, \mathbf{x})$ を隠れ状態 h_i と h_t を用いて計算する。この計算は全ての x_i, x_t の対に対して行われる。図 4 における**親への注意機構**中の弧は、それぞれの子に対し、親となる確率がもっとも高い単語を示している。

2. **再帰的注意機構**は x_t の d 次の親が x_j となる確率 $\alpha_{d,t,j}$ を、再帰的に $P_{parent}(x_i|x_t, \mathbf{x})$ を用いることで計算する。$\alpha_{d,t,j}$ は d 次の親を考慮した際の隠れ状態 $\mathbf{h}$ の重み付き和 $\gamma_{d,t}$ を計算するための注意として扱われる。図 4 において、単語 x_7 の親となる確率がもっとも高い単語を 3 次まで辿った場合、単語 x_6、x_5、x_2 がそれぞれ 1 次、2 次、3 次の単語 x_7 の親となる。これらの親に対応する隠れ状態 h_6、h_5、h_2 はそれぞれ 1 次、2 次、3 次の注意 $\alpha_{1,7,6}$、$\alpha_{2,7,5}$、$\alpha_{3,7,2}$ によって重み付けされた後、足し合わされ、選択的注意機構の入力 $\gamma_{d,t}$ となる。

3. **選択的注意機構**は各 $\gamma_{d,t}$ の次数 $d \in \mathbf{d}$ に対する重み $\beta_{d,t}$ を計算する。$\mathbf{d}$ は次数の集合を表す。$\beta_{d,t}$ はエンコーダとデコーダの隠れ状態から計算される。それぞれの $\beta_{d,t} \cdot \gamma_{d,t}$ の総和 Ω_t が**選択的注意機構**の出力となる。

4. 出力層において Ω_t とエンコーダ、デコーダの隠れ状態が結合され、各ラベルの出力確率が計算される。

各注意機構の詳細と目的関数の詳細については以下の節で説明される。

3.1 親への注意機構

提案手法は、Zhang ら [12] や Hashimoto ら [5] と同じく、依存構造を注意機構上のグラフとして表現している。依存構造木では子となる単語は 1 つの親となる単語を持つ。この制約の下で、依存構造解析は次のように表される。文 $\mathbf{x} = (x_0, x_1, ..., x_n)$ が与えられた時、$t \neq 0$ を満たす単語 x_t の親は、$i \neq t$ を満たす単語 x_i から選択される。なお x_0 はルートノードを表す。x_t の親が x_i となる確率は重み行列 W_g、v_a、U_a、W_a を用いて

$$P_{parent}(x_i|x_t, \mathbf{x}) = softmax(W_g \cdot g(h_j, h_t)) \cdot \delta_{x_j}, \quad (3)$$

$$g(h_j, h_t) = v_a^T \cdot tanh(U_a \cdot h_j + W_a \cdot h_t), \quad (4)$$

のように計算される。構文解析器とは異なり、提案手法では依存構造木を確定することはせず、$P_{parent}(x_i|x_t, \mathbf{x})$ をそのまま使用している。

3.2 再帰的注意機構

再帰的注意機構は、x_t の d 次の親が x_j となる確率を

$$\alpha_{d,t,j} = \begin{cases} \sum_{k=1}^{n} \alpha_{d-1,t,k} \cdot \alpha_{1,k,j} & (d>1) \\ P_{parent}(x_i|x_t, \mathbf{x}) & (d=1) \end{cases} \quad (5)$$

のように再帰的に計算する。また、ルートノードは親を持たず、子と親が同じ単語になることはないという依存構造木の制約を注意機構上に取り込むために、我々は $\alpha_{1,t,j}$ に対し、

$$\alpha_{1,t,j} = \begin{cases} 1 & (t = 0 \wedge j = 0) \\ 0 & (t = 0 \wedge j > 0) \\ 0 & (t \neq 0 \wedge t = j) \end{cases} \quad (6)$$

のような制約を課した。式 (6) の 1 行目と 2 行目よりルートノードの親はルートノードとなる。これらの制約はルートノードが親を持たないということを表している。式 (6) の 3

		ALL					DEPTH				
		$\mathbf{F_1}$	ROUGE			ΔC	$\mathbf{F_1}$	ROUGE			ΔC
			1	*2*	*L*			*1*	*2*	*L*	
Tagger		82.8	81.1	72.4	80.9	-3.0	**82.8**	80.6	72.2	80.3	-3.2
Tagger+ILP		79.0	76.1	64.6	75.8	-4.1	77.5	74.7	64.1	74.3	-4.6
Bi-LSTM		81.9	81.1	73.7	80.9	-2.2	81.3	80.4	73.3	80.1	-2.2
Bi-LSTM-Dep		82.3	81.5	74.1	81.3	-2.1	81.5	80.7	73.5	80.3	-2.1
Attn		82.4	81.6	74.3	81.4	-2.3	81.9	81.0	73.9	80.6	-2.3
Base		82.7	81.9	74.7	81.7	-2.4	82.1	81.0	73.9	80.7	-2.5
Prop	($\mathbf{d} = \{1\}$)	83.0	81.7	74.5	81.5	-2.9	82.7	81.4	74.5	81.1	-2.8
Prop	($\mathbf{d} = \{1, 2\}$)	83.1	82.5	75.2	82.2	-2.1	82.6	81.8	74.9	81.5	-2.1
Prop	($\mathbf{d} = \{1, 2, 4\}$)	**83.2**	**82.9**	**75.8**	**82.7**	**−1.7**	**82.8**	**82.2**	**75.5**	**82.0**	**−2.0**

表 1: 自動評価の結果。 **ALL** と **DEPTH** は、 それぞれ全ての文での評価と、深い木 (深さがテストデータ全体での平均値である 8 以上の木) を持つ文での評価結果を表している。$\mathbf{d}$ は使用した次数の集合を表している。

行目は親と子が同じ単語になることを防いでいる。式 (5) の 1 行目が行列積の定義を満たすために、この再帰的な注意の計算は CPU 及び GPU 上において効率的に計算することが可能である。

このように再帰的に注意を用いることによって、高次の親の確率を計算する際に、それぞれの親ごとに注意機構を準備する必要がなくなる。また、それにより、複数の注意を学習する必要がなくなるため、学習時に、各注意の分布の重みを調整するためのハイパーパラメータも同様に不要となる。その上、この手法により高次の依存関係を直接考慮する必要がなくなるため、学習データがスパースになることを防ぐことができる。

上記で計算された $\alpha_{d,t,j}$ は以下のように、

$$\gamma_t^d = \sum_{k=1}^{n} \alpha_{d,t,k} \cdot h_k, \tag{7}$$

エンコーダの両方向 LSTM の隠れ状態 $\mathbf{h}$ に対する重み付き和の計算に用いられる。計算された γ_t^d は**選択的注意機構**の入力に引き継がれる。

3.3 選択的注意機構

入力文に対し、適切な次数の依存関係を考慮するために、選択的注意機構は各隠れ状態 γ_t^d の重み付け和 Ω_t を、

$$\beta_{d,t} = softmax(W_c \cdot c_t) \cdot \delta_d, \tag{8}$$

$$\Omega_t = \sum_{k \in \{0\} \cup \mathbf{d}} \beta_{k,t} \cdot \gamma_{k,t}, \tag{9}$$

のように計算する。W_c はソフトマックス層の重み行列を、$\mathbf{d}$ は次数の集合を、c_t は現在の文脈をそれぞれ表す。なお、$\gamma_{0,t}$ はゼロベクトルであり、$\beta_{0,t}$ は依存構造木の情報を使用しないことを意味する。また、文脈ベクトル c_t は $c_t = [\overleftarrow{h}_0, \overrightarrow{h}_n, \overrightarrow{s}_t]$ のように計算される。

上記のように計算された Ω_t は出力層の入力に結合される。具体的には式 (1) の d_t は結合後のベクトル $d'_t = [h_t, \Omega_t, \overrightarrow{s}_t]$ に置き換えられる。またデコーダの次の入力に与えられる d_t も同様に d'_t に置き換えられる。

3.4 目的関数

構文解析誤りの影響を低減するために、提案手法は 1 次の依存関係を表す注意の分布 $\alpha_{1,t,k}$ と、ラベルの出力確率 $P(\mathbf{y}|\mathbf{x})$ を同時に学習する。$\alpha_{1,t,k}$ は既存の構文解析器の結果を用いて学習される。訓練データ中の依存構造木において、単語 w_t の親が w_j であることを $a_{t,j} = 1$ と定義し、単語 w_t の親が w_j ではないことを $a_{t,j} = 0$ と定義した時、提案手法の目的関数は、

$$-logP(\mathbf{y}|\mathbf{x}) - \lambda \cdot \sum_{i=1}^{n} \sum_{t=1}^{m} a_{t,j} \cdot log\alpha_{1,t,j}, \tag{10}$$

と定義される。λ は出力ラベルと依存構造解析の重要度を調整するためのハイパーパラメータである。

4 実験

評価には Google 文圧縮データセットを使用した。テストデータは従来研究と同様に評価セットの先頭 1000 文を用いた。訓練データには公開されている 20 万文のデータセットを使用した。訓練データ中に 10 回未満しか出現しない単語は未知語として扱った。品詞、依存構造ラベル、依存構造木解については、データセットに含まれている依存構造解析器による出力を利用した。

実験に際し、提案手法を以下のベースラインと比較した。なお、公平な比較を行うために、全てのベースラインに対して第 2 節で説明した入力ベクトルを使用した。

- **Tagger:** 両方向 LSTM とソフトマックス層により構成され、ラベルに対する点推定を行う手法 [8]。
- **Tagger+ILP:** 整数計画問題に基づく制約上で、**Tagger** の出力スコアを用いて単語の削除を決定する手法 [11]。
- **Bi-LSTM:** Fillipova ら [2] の構文解析結果を用いない手法のエンコーダを、公平な比較のために両方向 LSTM に置き換えた手法。片方向のそれぞれ LSTM の最終状態の和がデコーダの初期状態へと引き継がれる。
- **Bi-LSTM-Dep:** Fillipova ら [2] の構文解析結果を用いる手法 (LSTM-PAR-PRES) のエンコーダを、**Bi-LSTM** と同じく、両方向 LSTM に置き換えた手法。
- **Attn:** 注意機構を持つ Seq2Seq [10] (concat attention)。
- **Base:** 第 2 節で説明した、Seq2Seq に基づく手法。

従来研究 [11] に従い、単語埋め込み、LSTM の隠れ層、注意ベクトルを 100 次元に設定した。全ての手法で LSTM の隠

Input	Pakistan signed a resolution on Monday to import 1,300 MW of electricity from Kyrgyz Republic and Tajikistan to overcome power shortage in summer season , said an official press release .
Gold	Pakistan signed a resolution to import 1,300 MW of electricity from Kyrgyz Republic and Tajikistan .
Tagger	Pakistan signed a resolution to import 1,300 MW of electricity Tajikistan to overcome shortage .
Tagger-ILP	Pakistan signed resolution to import MW said .
Base	Pakistan signed a resolution to import 1,300 MW of electricity .
Prop (d=$\{1, 2, 4\}$)	Pakistan signed a resolution to import 1,300 MW of electricity from Kyrgyz Republic and Tajikistan .
Input	US whistleblower Bradley Manning , charged with releasing over 700,000 battlefield reports from Iraq and Afghanistan to Wikileaks , received a sentence of 35 years in prison from a military court Wednesday .
Gold	Bradley Manning received a sentence of 35 years in prison .
Tagger	Bradley Manning received a sentence of 35 years .
Tagger-ILP	Bradley Manning received a sentence of years .
Base	Bradley Manning received a sentence of 35 years .
Prop (d=$\{1, 2, 4\}$)	Bradley Manning received a sentence of 35 years in prison .

表 2: テストデータ中に存在する従来手法と提案手法の圧縮文の例.

れ層のサイズを同じ値にするために、**Tagger** を用いる手法では 3 層を、デコーダを用いる手法では 2 層を用いた。この設定により、全ての手法が 6 層の LSTM を持つことになる。品詞と依存関係ラベルの埋め込みには 40 次元を使用した。重みの初期値については、-1 から 1 の範囲で一様分布からランダムにサンプリングした [4]。また各 LSTM 層の入力ではドロップアウトを行い、その比率は 0.3 に設定した。

学習には Adam [7] を使用し、学習率の初期値は 0.001 に設定した。学習は訓練データ全体に対し、30 回行った。同時学習のための λ は 1.0 に設定した。ミニバッチのサイズは 16 に設定し、ミニバッチごとに誤差を平均した。なお、ミニバッチの順序は学習時にランダムに入れ替えた。clipping threshold は 5.0 に設定した。実装には Dynet を用いた。

評価にはトークン単位の F 値と、ROUGE-1,2,L を使用した。また、適切な長さが学習されていることを確認するために、ΔC = システムの圧縮率 − 正解の圧縮率 をトークン単位で計算して評価した。さらに、深い依存構造木を持つ文における性能を評価するために、依存構造木の深さがテストデータ全体での平均値である 8 以上の文だけでの評価も実施した。また、初期値による値の揺れを考慮し、全てのスコアを 5 回の試行の平均値とした。

表 1 に実験結果を示す。提案手法 Prop ($d = \{1, 2, 4\}$) はベースライン手法に比べほぼ全てのスコアで改善が見られる。特に F 値では従来の.82 [2] を上回る、このデータセットにおける最高精度を達成している。また、**DEPTH** では、特に ROUGE スコアにおいて **ALL** よりも大きな改善が確認されることから、提案手法は深い依存構造木を持つ文に対して有効であることが示された。

表 2 に、テストデータ中の実際の圧縮文を示す。出力例より、**Base** の出力は可読性は高いが、重要箇所が欠落してしまっていることが分かる。また、**Tagger** の出力については、文意が保存されておらず、また重要箇所も欠落してしまっている。さらに、**Tagger-ILP** では、葉ノードの情報が優先的に削除されるという特徴のために、数字の欠落が生じている。これらの結果から、提案手法は可読性を保持しつつ、重要箇所を圧縮文に残すことが可能であることが分かる。

5 結論

本稿では、深い依存構造木を持つ文を高精度に圧縮するために、高次の依存関係を考慮可能な注意機構を Seq2Seq に組み込む手法を提案した。Google 文圧縮データセットを用いた実験の結果、提案手法は従来手法に比して高精度に文を圧縮可能であることが示された。特に深い依存構造木を持つ文において改善の傾向はより顕著であった。以上より高次の依存関係を考慮可能な注意機構を用いることによって、深い依存構造木を持つ文において文圧縮の性能が向上することが示された。

参考文献

[1] Taylor Berg-Kirkpatrick, Dan Gillick, and Dan Klein. Jointly learning to extract and compress. In *Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies*, pp. 481–490, Portland, Oregon, USA, June 2011. Association for Computational Linguistics.

[2] Katja Filippova, Enrique Alfonseca, Carlos A. Colmenares, Lukasz Kaiser, and Oriol Vinyals. Sentence compression by deletion with lstms. In *Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing*, pp. 360–368, Lisbon, Portugal, September 2015. Association for Computational Linguistics.

[3] Katja Filippova and Yasemin Altun. Overcoming the lack of parallel data in sentence compression. In *Proceedings of the 2013 Conference on Empirical Methods in Natural Language Processing*, pp. 1481–1491, Seattle, Washington, USA, October 2013. Association for Computational Linguistics.

[4] Xavier Glorot and Yoshua Bengio. Understanding the difficulty of training deep feedforward neural networks. In *Proceedings of the Thirteenth International Conference on Artificial Intelligence and Statistics*, pp. 249–256, 2010.

[5] Kazuma Hashimoto and Yoshimasa Tsuruoka. Neural machine translation with source-side latent graph parsing. In *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing*, pp. 125–135, Copenhagen, Denmark, September 2017. Association for Computational Linguistics.

[6] Hongyan Jing. Sentence reduction for automatic text summarization. In *Proceedings of the Sixth Conference on Applied Natural Language Processing*, pp. 310–315, Seattle, Washington, USA, April 2000. Association for Computational Linguistics.

[7] Diederik P. Kingma and Jimmy Ba. Adam: A method for stochastic optimization. *CoRR*, Vol. abs/1412.6980, , 2014.

[8] Sigrid Klerke, Yoav Goldberg, and Anders Søgaard. Improving sentence compression by learning to predict gaze. In *Proceedings of the 2016 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pp. 1528–1533, San Diego, California, June 2016. Association for Computational Linguistics.

[9] Kevin Knight and Daniel Marcu. Statistics-based summarization-step one: Sentence compression. *AAAI/IAAI*, Vol. 2000, pp. 703–710, 2000.

[10] Thang Luong, Hieu Pham, and Christopher D. Manning. Effective approaches to attention-based neural machine translation. In *Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing*, pp. 1412–1421, Lisbon, Portugal, September 2015. Association for Computational Linguistics.

[11] Liangguo Wang, Jing Jiang, Hai Leong Chieu, Chen Hui Ong, Dandan Song, and Lejian Liao. Can syntax help? improving an lstm-based sentence compression model for new domains. In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pp. 1385–1393, Vancouver, Canada, July 2017. Association for Computational Linguistics.

[12] Xingxing Zhang, Jianpeng Cheng, and Mirella Lapata. Dependency parsing as head selection. In *Proceedings of the 15th Conference of the European Chapter of the Association for Computational Linguistics: Volume 1, Long Papers*, pp. 665–676, Valencia, Spain, April 2017. Association for Computational Linguistics.