両方向の合意制約を用いた ニューラルネットワークによる単語アライメント

田村 晃裕　　渡辺 太郎　　隅田 英一郎
(独) 情報通信研究機構
{akihiro.tamura, taro.watanabe, eiichiro.sumita}@nict.go.jp

1　はじめに

対訳文内で各単語の対応関係を特定する単語アライメントは，統計的機械翻訳に欠かせない重要な処理の一つである．単語アライメント手法の中で，最も著名な手法は，生成モデルである IBM モデル 1-5[1] 及び HMM モデル [11] であり，それらを拡張した手法が数多く提案されている．近年では，Yang ら [12] が，音声認識や統計的機械翻訳を含む多くの自然言語処理で成果をあげているニューラルネットワーク（NN）を HMM モデルに適用した手法を提案し，中英アライメントタスクにおいて IBM モデル 4 や HMM モデルよりも高い性能を実現している．

Yang らのモデルは，方向性（原言語 $\boldsymbol{f}$ から目的言語 $\boldsymbol{e}$ あるいは $\boldsymbol{e}$ から $\boldsymbol{f}$）を持っており，各方向のモデルは独立に学習，使用される．一方で，方向性を持つモデルにおいて，両方向の合意を取るように，それらのモデルを同時に学習することで，アライメント性能を改善できることが示されている [3, 5, 7, 8]．そこで，本稿では，Yang らの NN ベースアライメントモデルに合意制約を導入した手法を提案する．両方向の合意は，両方向の word embedding を一致させるようにモデルを学習することで実現する．具体的には，両方向の word embedding の差を表すペナルティ項を目的関数に導入する．そして，日英及び仏英単語アライメントタスクの実験を通じて，合意制約を導入することによりアライメント性能を改善できることを示す．

2　HMM モデル

HMM モデルを含めた生成モデルでは，原言語の文 $f_1^J = f_1, ..., f_J$ とそれに対応する目的言語の文 $e_1^I = e_1, ..., e_I$ がある時，f_1^J は e_1^I からアライメント $a_1^J = a_1, ..., a_J$ を通じて生成されると考える[1]．ここで，各 a_j は，f_j が e_{a_j} に対応する事を示す隠れ変数である．そして，f_1^J の生成確率は次の通り定義される：

$$p(f_1^J|e_1^I) = \sum_{a_1^J} p(f_1^J, a_1^J|e_1^I). \quad (1)$$

HMM モデルは，式 (1) をアライメント確率 p_a と語彙翻訳確率 p_t に分解する：

$$p(f_1^J, a_1^J|e_1^I) = \prod_{j=1}^{J} p_a(a_j|a_j - a_{j-1}) p_t(f_j|e_{a_j}). \quad (2)$$

このモデルは，EM アルゴリズムにより，対訳コーパスから学習する．また，対訳文対 (f_1^J, e_1^I) に対して，次式 (3) を満たす最適なアライメント（ビタビアライメント）は，学習したモデルを用いて，forward-backward アルゴリズムにより決定する：

$$\hat{a}_1^J = \underset{a_1^J}{\operatorname{argmax}}\, p(f_1^J, a_1^J|e_1^I). \quad (3)$$

3　NN ベースモデル

本節では，提案手法のベースラインとなる Yang らの手法 [12] を説明する．この手法は，NN の一種である「Context-Dependent Deep Neural Network for HMM」[2] を HMM アライメントモデルに適用した手法である．具体的には，式 (2) の p_a 及び p_t をフィードフォワード型 NN を用いて計算する：

$$s_{NN}(a_1^J|f_1^J, e_1^I) = \prod_{j=1}^{J} t_a(a_j - a_{j-1}|c(e_{a_{j-1}})) \cdot t_t(f_j, e_{a_j}|c(f_j), c(e_{a_j})). \quad (4)$$

[1]通常，f_j がどの目的言語の単語にも対応しない場合を扱うために，単語「null」(e_0) が目的言語の文に加えられるが，本稿では簡単のために割愛する．

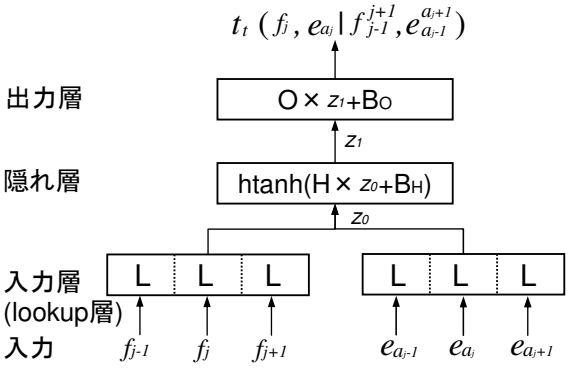

図 1: 語彙翻訳スコア $t_t(f_j, e_{a_j})$ 計算用ネットワーク

ただし，全単語における正規化は計算量が膨大となるため，確率の代わりにスコアを用いる．t_a 及び t_t は，それぞれ，p_a と p_t に対応する．また，s_{NN} はアライメント a_1^J のスコアであり，「c(単語 w)」は単語 w の文脈を表す．ビタビアライメントは，本モデルにおいても HMM モデル同様，forward-backward アルゴリズムにより決定する．

図 1 に，語彙翻訳スコア $t_t(f_j, e_{a_j}|c(f_j), c(e_{a_j}))$ を計算するネットワーク構造（語彙翻訳モデル）を示す．このネットワークは，lookup 層（入力層），隠れ層，出力層から構成され，各層は，それぞれ，重み行列 L，$\{H, B_H\}$，$\{O, B_O\}$ を持つ．L は，embedding 行列であり，各単語の word embedding を管理する．word embedding とは，単語を特徴付ける低次元の実ベクトルであり，単語の統語的，意味的特性を表す．原言語の単語集合を V_f，目的言語の単語集合を V_e，word embedding の長さを M とすると，L は $M \times (|V_f| + |V_e|)$ 行列である．ただし，V_f と V_e には，それぞれ，未知語を表す $\langle unk \rangle$ と単語「null」を表す $\langle null \rangle$ を追加する．

この語彙翻訳モデルは，入力として，計算対象である原言語の単語 f_j と目的言語の単語 e_{a_j} と共に，それらの文脈単語を受け付ける．文脈単語とは，予め定めたサイズの窓内に存在する単語であり，図 1 は窓幅が 3 の場合である．まず，lookup 層が，入力の各単語に対して embedding 行列 (L) から対応する列を見つける（word embedding を割り当てる）．そして，それらを結合させた実ベクトル (z_0) を隠れ層に送る．次に，隠れ層が，z_0 の非線形な特徴を捉える．最後に，出力層が，隠れ層の出力 (z_1) を受け取り，語彙翻訳スコアを計算して出力する．隠れ層，出力層が行う具体的な計算は以下の通りである[2]：

$$z_1 = \text{htanh}(H \times z_0 + B_H), \tag{5}$$

$$t_t = O \times z_1 + B_O. \tag{6}$$

ここで，H，B_H，O，B_O は，それぞれ，$|z_1| \times |z_0|$，$|z_1| \times 1$，$1 \times |z_1|$，1×1 行列である．また，非線形活性化関数として htanh(x)[3]を用いている．

アライメントスコア $t_a(a_j - a_{j-1}|c(e_{a_{j-1}}))$ を計算するアライメントモデルも，語彙翻訳モデルと同様に構成できる．各モデルの学習では，次式 (7) のランキング損失を最小化するように，各層の重み行列を確率的勾配降下法により学習する．各重みの勾配はバックプロパゲーションで計算する：

$$loss(\theta) = \sum_{(\boldsymbol{f},\boldsymbol{e}) \in T} \max\{0, 1 - s_\theta(\boldsymbol{a}^+|\boldsymbol{f},\boldsymbol{e}) + s_\theta(\boldsymbol{a}^-|\boldsymbol{f},\boldsymbol{e})\}. \tag{7}$$

ここで，θ は最適化するパラメータ（重み行列の重み），T は学習データ，s_θ はパラメータ θ のモデルによる a_1^J のスコア（式 (4) 参照），$\boldsymbol{a}^+$ は正解アライメント，$\boldsymbol{a}^-$ はパラメータ θ のモデルでスコアが最高の不正解アライメントを示す．

4 合意制約の導入

3 節の NN ベースモデルは，多くのアライメントモデル同様，方向性を持つモデルである．すなわち，$\boldsymbol{f}$ に対して $\boldsymbol{e}$ とのアライメントをモデル化することにより，単語 f_j の $\boldsymbol{e}$ との 1 対多アライメントを表す．通常，これらのモデルは方向毎に独立に学習されるが，両方向のモデルの合意を取るように同時に学習することで，性能を改善できることが示されている．これは，学習される特徴や汎化性は方向毎に異なり，それらは相補的であるとの考えに基づいている．例えば，Matusov ら [8] や Liang ら [7] は，両方向のモデルパラメータで定義した目的関数を使い，両方向のモデルを同時に学習している．また，Ganchev ら [3] や Graça ら [5] は，EM アルゴリズムの E ステップで，モデルパラメータの事後分布に対して合意制約を課している．

そこで，本節では，NN ベースモデルの学習に合意制約を導入する．具体的には，両方向の word embedding

[2]本稿の隠れ層は 1 層であるが，連続した l 層の隠れ層を用いる事もできる：$z_l = f(H_l \times z_{l-1} + B_{H_l})$．複数の隠れ層を用いた実験は今後の課題とする．

[3]$x < -1$ の時，htanh$(x) = -1$，$x > 1$ の時，htanh$(x) = 1$，それ以外の時，htanh$(x) = x$ である．

が一致するようにモデルを学習する．これを実現するために，二つの word embedding の差を表すペナルティ項を目的関数に導入し，その目的関数（次式 (8)，(9)）に基づいて両方向のモデルを同時に学習する：

$$\underset{\theta_{FE}}{\operatorname{argmin}}\big\{loss(\theta_{FE}) + \alpha\|\theta_{L_{EF}} - \theta_{L_{FE}}\|\big\}, \quad (8)$$

$$\underset{\theta_{EF}}{\operatorname{argmin}}\big\{loss(\theta_{EF}) + \alpha\|\theta_{L_{FE}} - \theta_{L_{EF}}\|\big\}. \quad (9)$$

ここで，θ_{FE} と θ_{EF} は，それぞれ，$\boldsymbol{f} \rightarrow \boldsymbol{e}$ と $\boldsymbol{e} \rightarrow \boldsymbol{f}$ のアライメントモデルのパラメータ，θ_L は lookup 層のパラメータ（L の重みであり word embedding を表す），α は合意制約の強さを制御するパラメータ，$loss(\theta)$ は式 (7) で定義されるランキング損失である．また，$\|\theta\|$ は θ のノルムを表す．実験では 2-ノルムを用いた．

5 実験

5.1 実験設定

提案手法の有効性を検討するため，二つのアライメントタスクおけるアライメント性能を評価する．Basic Travel Expression Corpus[10] における日英の単語アライメントタスク（*BTEC*）と，NAACL 2003 の shared task で使われた Hansards データにおける仏英の単語アライメントタスク（*Hansards*）である．*BTEC* おける学習データは 9K，テストデータは 960 の対訳文対である．また，*Hansards* における学習データは 100K[4]，テストデータは 447 である．*BTEC* の学習データには，正解の単語アライメントが人手で付与されている [4]．一方で，*Hansards* の学習データには，単語アライメントは付与されていない．

実験では，合意制約を導入した NN ベースモデル (*NN*+*c*) の性能に加え，ベースラインとして合意制約を使わない NN ベースモデル (*NN*) と，最も一般的に使われている IBM モデル 4($IBM4$) の性能を評価した．$IBM4$ は，IBM モデル 1-4 と HMM モデルを順番に適用して学習した [9]：$1^5H^53^54^5$．NN ベースモデル (*NN* と *NN*+*c*) では，word embedding の長さ M を 30，文脈の窓幅を 5 とした．また，隠れ層として，ユニット数 $|z_1|$ が 100 の層を 1 層使用した．NN ベースモデルの学習では，まず，各層の重みを初期化する．lookup 層の重みは，学習データの原言語側，目的言語側からそれぞれ予め学習した word embedding に設定する．word embedding の学習には，RNNLM Toolkit[5]（デフォルトのオプション）を用いた．その際，コーパスでの出現数が 5 回以下の単語は $\langle unk \rangle$ に置き換えた．その他の層の重みは，無作為に $[-0.1, 0.1]$ の値に設定する．その後，各重みを，式 (7) あるいは式 (8)，(9) を目的関数として，確率的勾配降下法によりミニバッチ学習する．本実験では，バッチサイズを 100，学習率を 0.01 とし，50 エポックで学習を終えた．また，学習データへの過学習を避けるため，目的関数には $l2$ 正則化項（正則化の比率は 0.1）を加えた．*NN*+*c* における合意制約のパラメータ α は 0.1 とした．

[4] shared task オリジナルの学習データの総数は約 1.1M であるが，実験では，学習時の計算量を削減するため，無作為にサンプリングした 100K を用いた．大規模データの実験は今後の課題とする．

Alignment	*BTEC*	*Hansards*
IBM4	48.59	90.29
NN(*IBM4*)	47.70	90.20
NN+*c*(*IBM4*)	48.54*	90.85*
NN(REF)	82.24	-
NN+*c*(REF)	83.67*	-

表 1: アライメント性能 (F1 値:%)

5.2 実験結果

表 1 に各手法のアライメント性能を示す．教師ありモデルである NN ベースモデルに対しては，学習データに付与されている正しいアライメントを学習したモデル (「モデル (REF)」と記す）と，*IBM4* で特定したアライメントを学習したモデル（「モデル (*IBM4*)」と記す）の二種類の性能を示す．*Hansards* の学習データには単語アライメントが付与されていないため,「モデル (REF)」は実現できないことを確認しておく．

評価手順は，まず，各アライメントモデルにより，$\boldsymbol{f} \rightarrow \boldsymbol{e}$ と $\boldsymbol{e} \rightarrow \boldsymbol{f}$ のアライメントをそれぞれ生成する．その後,「grow-diag-final-and」ヒューリスティクス [6] を用いて，両方向のアライメントを結合する．そして，その結合したアライメント結果を，F1 値で評価する．また，有意差検定は，有意差水準 5%の符号検定で行う．表 1 中の「*」は，対応するベースライン *NN*(*IBM4*/REF) との性能差が有意であることを示す．

表 1 より，*BTEC* と *Hansards* のどちらのタスクにおいても，*NN*+*c* は *NN* よりも有意にアライメント性能が良い．この結果より，NN ベースモデルにおいて，合意制約を導入することでアライメント性能を改善できることが実験的に確認できる．また，*BTEC* において，合意制約の導入の効果は，*NN*(REF) の方が *NN*(*IBM4*) よ

[5] http://www.fit.vutbr.cz/~imikolov/rnnlm/

り大きいことが分かる（*NN*(REF)→*NN*+*c*(REF):+1.43, *NN*(*IBM4*)→*NN*+*c*(*IBM4*):+0.84）．これは，*IBM4* が特定したアライメントには誤りが多く含まれているため，合意制約を導入し，そのアライメントを正しく学習できるようになったとしても，アライメント性能の改善に直結しない場合があることが原因の一つと考えられる．

また，合意制約の導入の効果は，日英（*BTEC*）の方が仏英 (*Hansards*) よりも大きい（*BTEC*:+1.43, +0.84, *Hansards*:+0.65）．これは，日英の方が言語間の違いが大きい（例えば，仏英の方が 1 対 1 アライメントが多い）ため，日英の方が，反対方向のアライメントを考慮することで新規に得られる情報が多い可能性を示している．

最後に，*NN*+*c*(*IBM4*) は，*BTEC* では *IBM4* と同等の性能であり，*Hansards* では *IBM4* より性能が良い．同様の事は，Yang ら [12] の実験において，言語対が中英の場合で確認されているが，今回，日英及び仏英のアライメントにおいても確認できた．

6 おわりに

本稿では，NN ベースアライメントモデル [12] に合意制約を導入した．具体的には，両方向の word embedding の差を表すペナルティ項を目的関数に導入することで，合意制約を課したモデルの学習を行う．日英及び仏英単語アライメントタスクの実験を通じて，合意制約を導入することによりアライメント性能を改善できることを示した．

今後は，合意制約によるアライメント性能の改善が翻訳性能の改善に寄与するかを調べる予定である．また，アライメント性能を更に向上させるため，Yang ら [12] のように複数の隠れ層を用いることも検討したい．

参考文献

[1] Peter F. Brown, Stephen A. Della Pietra, Vincent J. Della Pietra, and Robert L. Mercer. The Mathematics of Statistical Machine Translation: Parameter Estimation. *Computational Linguistics*, Vol. 19, No. 2, pp. 263–311, 1993.

[2] George E. Dahl, Dong Yu, Li Deng, and Alex Acero. Context-Dependent Pre-trained Deep Neural Networks for Large Vocabulary Speech Recognition. *Audio, Speech, and Language Processing, IEEE Transactions on*, Vol. 20, No. 1, pp. 30–42, 2012.

[3] Kuzman Ganchev, João V. Graça, and Ben Taskar. Better Alignments = Better Translations? In *Proc. ACL/HLT 2008*, pp. 986–993, 2008.

[4] Chooi-Ling Goh, Taro Watanabe, Hirofumi Yamamoto, and Eiichiro Sumita. Constraining a Generative Word Alignment Model with Discriminative Output. *IEICE Transactions*, Vol. 93-D, No. 7, pp. 1976–1983, 2010.

[5] João V. Graça, Kuzman Ganchev, and Ben Taskar. Expectation Maximization and Posterior Constraints. In *Advances in NIPS 20*, pp. 569–576, 2008.

[6] Philipp Koehn, Franz Josef Och, and Daniel Marcu. Statistical Phrase-Based Translation. In *Proc. HLT/NAACL 2003*, pp. 48–54, 2003.

[7] Percy Liang, Ben Taskar, and Dan Klein. Alignment by Agreement. In *Proc. HLT/NAACL 2006*, pp. 104–111, 2006.

[8] Evgeny Matusov, Richard Zens, and Hermann Ney. Symmetric Word Alignments for Statistical Machine Translation. In *Proc. Coling 2004*, pp. 219–225, 2004.

[9] Franz Josef Och and Hermann Ney. A Systematic Comparison of Various Statistical Alignment Models. *Computational Linguistics*, Vol. 29, pp. 19–51, 2003.

[10] Toshiyuki Takezawa, Eiichiro Sumita, Fumiaki Sugaya, Hirofumi Yamamoto, and Seiichi Yamamoto. Toward a Broad-coverage Bilingual Corpus for Speech Translation of Travel Conversations in the Real World. In *Proc. LREC 2002*, pp. 147–152, 2002.

[11] Stephan Vogel, Hermann Ney, and Christoph Tillmann. Hmm-based Word Alignment in Statistical Translation. In *Proc. Coling 1996*, pp. 836–841, 1996.

[12] Nan Yang, Shujie Liu, Mu Li, Ming Zhou, and Nenghai Yu. Word Alignment Modeling with Context Dependent Deep Neural Network. In *Proc. ACL 2013*, pp. 166–175, 2013.