数式曖昧表記変換型数式入力の機械学習による数式予測と精度

福井哲夫
武庫川女子大学大学院生活環境学研究科
fukui@mukogawa-u.ac.jp

概要　本研究は，数式入力インタフェースに自然言語処理技術を応用し，インテリジェント化を試みたものである．従来，数式入力にはGUIを使い，分数や指数など，数式構造を表すアイコンテンプレートから選択して，要素を入力し数式を構築する構造ベース入力方式が代表的である．阪大の土方らはこの数式要素をN-gramモデルにより予測し効率を向上させている．本研究では，2011年に提案され，テキストベースの入力に近い，数式曖昧表記変換型数式入力の候補算出アルゴリズムに，構造化パーセプトロンによる機械学習を応用し，インテリジェントな数式予測が行えるようにした．スコア学習のパラメタや高速化のための探索的アルゴリズムによって予測精度が異なるため，最良パラメタについての検討を行った結果，一つのアルゴリズムは，変換候補ベスト1に対する正解率が約79.1 %で，ベスト１０まで含めると約89.2 %になった．一方，もう一つのアルゴリズムではベスト1が約68.5 %で，ベスト１０まで含めると95.0 %と，ほぼ実用レベルに近づいた．まだ，本研究は萌芽的で，学習データの数学分野が限定的ではあるが，これまでの研究結果を報告する．

キーワード　数式入力，機械学習，知的インタフェース，数式曖昧表記変換

1　はじめに

デジタル書籍・デジタル教科書の導入が進む中，数式を表記する仕組みは，LaTeXやMathML形式などを中心に発展してきている．しかし，ユーザによる数式のデジタル入力の方法は，依然わずらわしいものが多い[1].

筆者はより効率のよい数式入力方法を模索し，数式曖昧表記変換方式を提案してきた[2],[3]．数式曖昧表記変換方式による数式入力とは，数式のための文字列を変換して数式を対話的に構築しようとするもので，入力時の文字列が単純なため負担が軽く，数式要素を左から順に提示された候補の中から選択・確定していく特徴をもつ．この手順によって所望する式を確実に構築（達成）できることが示されている[4],[5].

しかし，上記の効果は演算子が作用するオペランドの範囲を正しく指示して確定していくことが前提であり，初心者がつまずく原因の一つでもある．また，構成要素の数だけ確定を繰り返す必要がある[6].

そこで，本研究のねらいは，この数式曖昧表記変換方式による数式入力において，候補の算出を数式要素毎ではなく，数式全体をインテリジェントに予測させ，選択・確定手順の効率を向上させることにある．

本論文では，数式曖昧表記変換方式候補算出アルゴリズムに構造化パーセプトロンによる機械学習技術を導入し、数式入力のインテリジェント化を行い，変換精度を評価することを目的とする．

2　先行研究

2.1　N グラムモデルによる数式予測入力

数式入力の代表的なインタフェースとして，GUIを使い，分数や指数など，数式構造を表すアイコンテンプレートから選択して，要素を入力し数式を構築する構造ベース入力方式がある．阪大の土方らはこの数式要素をN-gramモデルにより予測し効率を向上させている[7],[8]．しかし，入力したい数式の全体構造をユーザが把握して，構造テンプレートから指示する方式は依然変わっていない．

一方，本研究で提案する数式曖昧表記変換方式は，テキストベースの入力をインテリジェント化しようとする試みであり，構造ベース入力方式とはアプローチが大きく異なる．

2.2　従来の曖昧表記変換方式

2012年に筆者が提案した数式曖昧表記変換方式[2, 3, 4, 5][1]とは，初期入力文字列に表記されない区切り記号や演算子の指示が不要で，数式変換エンジンと数式辞書（変換キーと数式要素と順序の対応データ）によって，候補を算出し，WYSIWYGでインタラクティブに数式を構築する方式である．ここで初期入力文字列のルール（数式曖昧表記法）は，所望する数式に含まれる数式要素に対応するキー文字（列）を読む順番に区切りなく並べるだけである．一般に，所望する数式の要素を読む順番に並べたものを$(e_1, e_2, \cdots, e_N)$とし，e_i $(i = 1, \cdots, N)$に対応するキー文字（列）をk_iとすると，数式曖昧表記文字列は“$k_1k_2\cdots k_N$”で表される．例えば，$-\frac{\alpha^2}{3}$ (マイナス, アルファ,2 乗, 割る,3) のための文字列は “-a2/3”

[1]参考文献[2]ではインテリジェントな線形入力方式と呼んでいた．

となる．このように数式曖昧表記法は数学記号（イタリック体，ボールド体，ローマン体，ギリシャ文字）などがほぼアルファベット 1 文字に対応しており，数式構造を決める演算子（べき乗 [2] や割り算など）が作用するオペランド範囲の区切り記号（括弧やコンマなど）が不要なため，CAS コマンド形式に比べ短い文字数で済むという特徴をもつ．しかし，数式変換時に演算子のオペランド範囲をその都度指定する必要がある [6]．数式曖昧表記変換方式の操作の流れを図 1 に示す．例えば，$\frac{1}{a+2}$ を入力したい場合は，数式要素 $(e_1, e_2, \cdots, e_5)$ =(1, 割る,a, プラス,2) の数式曖昧表記文字列は “1/a+2” となり，これをテキストフィールドに入力して変換開始すると，“1”,“/”,“a”,“+”,“2”の順に WYSIWYG で変換候補が表示され（図 2），インタラクティブに確定していく．演算子の場合はそのオペランド範囲が赤のアンダーラインで強調表示され，左右矢印キーで調整することで，範囲を指定できる．全ての要素 $(e_1, e_2, \cdots, e_5)$ を確定したら変換操作は終了となる．

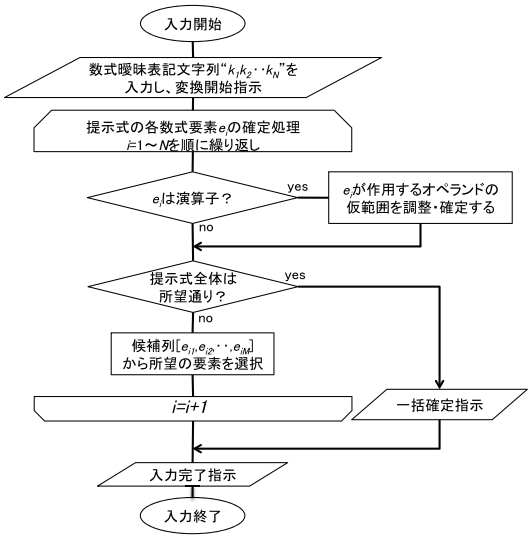

図 1 数式曖昧表記変換方式の入力操作手順

この方式を実装した数式入力インタフェースを MathTOUCH と呼んでいる [5]．MathTOUCH で扱える数式の範囲は，LaTeX で扱える数式をほぼカバーしており，大学の教養数学程度の数式は入力可能である．

3 インテリジェント化の設計

3.1 設計方針

本研究では，数式曖昧表記文字列 s を入力し，最適候補の数式 y_p を出力する問題を考える．そのため，数式の

[2] べき乗は 2 次元表記の数式に表記されない演算子として扱い，数式曖昧表記法では指示文字は不要（Maxima では “^” を使う）である．したがって，x^2 の場合は “x2” となる．このとき，変換候補列には $[x2, x^2, x_2, {}^x2, {}_x2]$ が挙がる．

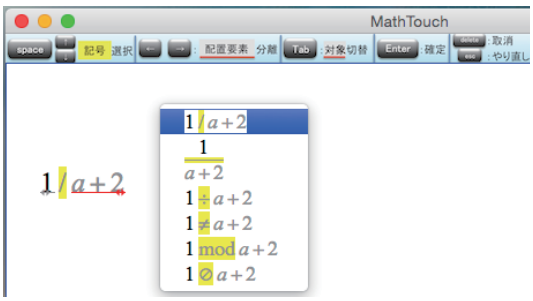

図 2 MathTOUCH による数式入力画面

出現頻度の高さを予測して候補を算出する必要がある．これにより，ユーザは数式要素毎ではなく，数式全体で候補を選ぶことができ，図 1 の繰返し効率が向上する．

出現頻度予測値として，数式を構成する要素などにスコアを付与し，それらのスコアを総合して数式のスコアを算出する方式を採用する．機械学習によって適切に調整されたスコア付与によって，数式 y に対するスコア算出関数 Score(y) を決定する．

入力された数式曖昧表記文字列 s から生成される可能な全ての数式全体の集合を $Y(s)$ とすれば，スコアが最大となる式 (1) の y_p が求める予測候補となる．

$$y_p \quad \text{s.t.} \quad \text{Score}(y_p) = \max\{\text{Score}(y) | y \in Y(s)\} \qquad (1)$$

3.2 機械学習アルゴリズムの設計

数式曖昧表記文字列 s から最適候補の数式 y_p を機械学習アルゴリズムによって算出するためには，3 つの項目：(1) 入力する特徴ベクトルの設計，(2) 予測関数の最適パラメタ決定アルゴリズム，(3)N ベスト算出アルゴリズムを検討する必要がある．

3.2.1 入力する特徴ベクトルの設計

本研究における，数式は数・変数などの数式最小単位記号自身であるか，または演算子 [3] としての構成要素（数式要素）と演算子が作用するオペランドへの接続関係から成る．このように数式は木構造を成しており，数式要素をノード，接続関係をエッジと呼ぶことがある．

数式要素には表 1 に示す 9 タイプがあり，数式曖昧表記変換方式において，あるタイプ（略号）t の数学記号 e がキー k から変換されとき，(k, e, t) によって特徴付けられる．例えば，2 の特徴は $(\text{“2”}, 2, N)$，x の特徴は $(\text{“x”}, x, V)$ であり，ギリシャ文字 α の特徴は $(\text{“alpha”}, \alpha, V)$ の場合と $(\text{“a”}, \alpha, V)$ の場合がある．演算子記号の場合も同じで，分数の特徴は $(\text{“/”}, \frac{\square_1}{\square_2}, C)$ となる．ここで，$\square_1, \square_2$ は仮のオペランドを表す．

一方，数式の接続関係は，親演算子 e_p が第 i 番目の

[3] ここで言う演算子とは，複数の部分式（オペランド）の配置に作用して，表記上の配置構造を決める数学記号のことであり，数学的計算を意味するものではない．

表 1 数式要素の 9 タイプ

数式要素タイプ	タイプ略号	例（$\square_1, \square_2, \square_3$ はオペランド）
数	N	$2, 128$
変数・シンボル	V	x, α
前置単項演算子	P	$\sqrt{\square_1}, \sin \square_1$
後置単項演算子	A	$\square_1'$
括弧ペア	B_L, B_R	$(\square_1)$
内挿 2 項演算子	C	$\square_1 + \square_2, \frac{\square_1}{\square_2}$
前置 2 項演算子	Q	$\log_{\square_1} \square_2$
前置 3 項演算子	R	$\int_{\square_1}^{\square_2} \square_3$
内挿 3 項演算子	T	$\square_1 \overset{\square_2}{\rightarrow} \square_3$

オペランドへの要素 e_c に接続するとき (e_p, i, e_c) として特徴付けられる．

例えば，数式 $\frac{x}{2}$ の場合は 3 つの数式要素の集合

$$\{e_1, e_2, e_3\} := \{(\text{“x”}, x, V), (\text{“/”}, \frac{\square_1}{\square_2}, C), (\text{“2”}, 2, N)\} \quad (2)$$

と，2 つの接続関係

$$\{(e_2, 1, e_1), (e_2, 2, e_3)\} \quad (3)$$

から成る．本研究の試作システムに実装した数式辞書では数以外の特徴は有限であり，数を除く数式最小単位記号の特徴は現時点で 510 個，演算子記号は 597 個定義されている．接続関係も 597 個の演算子を親として，それぞれが 1107 個のノードへと接続される特徴があり，数要素を含めた総数は膨大な数になる．この特徴総数を F_{total} で表し，F_{total} 次元ベクトル $\boldsymbol{x}$ を考え，数式 y が第 f 番目の特徴を含む場合，$\boldsymbol{x}$ の第 f 成分 $x_f = 1$（ただし，一つの数式に同じ要素が複数含まれる場合はその個数）とし，それ以外の成分を 0 として数式 y に含まれる全特徴をベクトルによって表現する．これが本研究で採用した数式の特徴ベクトルである．

$$\boldsymbol{x} = \begin{pmatrix} x_1 \\ x_2 \\ \vdots \\ x_{F_{total}} \end{pmatrix} \quad (4)$$

したがって，数式 $\frac{x}{2}$ の場合の特徴ベクトルは式 (2),(3) の 5 つの特徴に対応する成分のみが 1 のベクトルとなる．

3.2.2 予測関数の最適パラメタ決定アルゴリズム

数式曖昧表記文字列 s から生成（生成の仕方は 3.2.3 項で議論する）される可能な全ての数式全体の集合を $Y(s)$ とする．ある数式 $y \in Y(s)$ から決まる特徴ベクトルを $\boldsymbol{x}(y)$ で表す．

今，何らかの方法で F_{total} 個の各特徴の出現の高さをスコアで表し記録したものをベクトル $\boldsymbol{\theta}^T = (\theta_1, \cdots, \theta_{F_{total}})$ で表すと，数式 $y \in Y(s)$ のスコアは式 (5) で表される．

$$h_\theta(\boldsymbol{x}(y)) = \boldsymbol{\theta}^T \cdot \boldsymbol{x}(y) = \sum_{i=1}^{F_{total}} \theta_i x_i(y) \quad (5)$$

これはまさに線形回帰の予測関数に他ならない．それゆえ，スコアパラメタ $\boldsymbol{\theta}$ の最適値を求めるために m 個の訓練データ $\{(s_1, y_1), (s_2, y_2), \cdots, (s_m, y_m)\}$ による教師あり機械学習を利用する．本研究で採用した訓練アルゴリズムは，次の (Step 1)〜(Step 4) で表される．(Step 1) でパラメタ $\boldsymbol{\theta}$ を初期化し，(Step 2) でその時点の $\boldsymbol{\theta}$ による数式候補を算出する（詳細は 3.2.3 項）．$\boldsymbol{\theta}$ の最適化（訓練）は (Step 3) の様に行う．

(Step 1) $\boldsymbol{\theta} = \boldsymbol{0}$, $i = 1$

(Step 2) y_p s.t.
$h_\theta(\boldsymbol{x}(y_p)) = \max\{h_\theta(\boldsymbol{x}(y)) \,|\, y \in Y(s_i)\}$

(Step 3) if($y_p \neq y_i$) {

$$\begin{aligned} \theta_f &:= \theta_f + 1 \quad \text{for} \quad \{f \leq F_{total} | x_f(y_i) = 1\} \\ \theta_{\bar{f}} &:= \theta_{\bar{f}} - 1 \quad \text{for} \quad \{\bar{f} \leq F_{total} | x_{\bar{f}}(y_p) = 1\} \end{aligned} \quad (6)$$

}

(Step 4) if($i \leq m$){ i=i+1; Step 2 へ繰り返し.}
else { θ を出力して終了 }

このアルゴリズムはとても単純であるが，(Step 2) でパラメタの中間層処理を行っており，構造化パーセプトロンによる機械学習アルゴリズムとなっている [9].

3.2.3 N ベスト算出アルゴリズム

数式要素の特徴の全体集合を $D_{ic} := \{(k, v, t)\}$ とし，ノード辞書と呼ぶ．ここで，k はキー文字（列）, v は数式最小単位記号または演算子記号, t は数式要素のタイプを表す．また，数式曖昧表記文字列 s が K 個のキー文字（列）に分解 $s = k_1 \uplus k_2 \uplus \cdots k_K$（ただし，$(k_i, v_i, t_i) \in D_{ic}, i = 1, ..., K$）できるとき，ベクトル $\boldsymbol{k}(s) = (k_1, k_2, \cdots, k_K)$ を s から生成されるキーベクトルと呼ぶ．また，s から生成されるキーベクトル全体を $K(s)$ で表す．例えば，円周率 π は $(\text{“pi”}, \pi, V) \in D_{ic}$ であることから，$K(\text{“2pi”}) = \{(\text{“2”}, \text{“pi”}), (\text{“2”}, \text{“p”}, \text{“i”})\}$ となる．

次に，数式要素のタイプ（表 1）を成分とするベクトルを数式構造ベクトル $\boldsymbol{t}$ と呼び，バッカスの正規形式に習って，式 (7) で再帰的に定義する．

$$\begin{aligned} \boldsymbol{t} \quad := \quad & N|V|(P, \boldsymbol{t})|(\boldsymbol{t}, A)|(B_L, \boldsymbol{t}, B_R)| \\ & (\boldsymbol{t}, C, \boldsymbol{t})|(Q, \boldsymbol{t}, \boldsymbol{t})|(R, \boldsymbol{t}, \boldsymbol{t}, \boldsymbol{t})|(\boldsymbol{t}, T, \boldsymbol{t}, \boldsymbol{t}) \end{aligned} \quad (7)$$

さて，s から生成される一つのキーベクトル $\boldsymbol{k}(s)$ が得られると，数式辞書を使って構造解釈することによって，可能な数式構造ベクトルを生成できる．その数式構造ベクトル全体を $T(\boldsymbol{k}(s))$ とする．例えば，s_0 =“x/2+pi” の場合は，$\boldsymbol{k}(s_0) = (\text{“x”}, \text{“/”}, \text{“2”}, \text{“+”}, \text{“pi”})$ が得られ，

3つの数式構造ベクトルの集合（式 (8)）を得る．

$$T(\boldsymbol{k}(s_0)) = \{((V,C,N),C,V), (V,C,(N,C,V)), (V,C,(N,SP,(P,V)))\} \quad (8)$$

ここで SP は空白すなわち省略された積などを表す2項演算子であり，加算演算子 + が前置単項演算子と解釈された場合に挿入される．これらの数式構造ベクトルが解ると，ノード辞書 D_{ic} 内の各キー $k_i(i=1,...,K)$ にマッチする数式要素を検索して数式を構築することができる．例えば式 (8) の場合は，$T(\boldsymbol{k}(s_0)) = \{\boldsymbol{t}_1, \boldsymbol{t}_2, \boldsymbol{t}_3\}$ とおくと，$\boldsymbol{t}_1$ の数式構造からは $\frac{x}{2}+\pi$ などが，$\boldsymbol{t}_2$ からは $\frac{x}{2+\pi}$ などが，$\boldsymbol{t}_3$ からは $\frac{x}{2+\pi}$ などが候補の一つとして生成される．

さらに，一つの数式構造ベクトル $\boldsymbol{t} \in T(\boldsymbol{k}(s))$ から生成される可能な全ての数式全体を $W(\boldsymbol{t})$ で表すと，s から生成される可能な全ての数式全体 $Y(s)$ は式 (9) より得られる．

$$Y(s) = \bigcup_{\boldsymbol{k} \in K(s)} \left(\bigcup_{\boldsymbol{t} \in T(\boldsymbol{k})} W(\boldsymbol{t}) \right) \quad (9)$$

さて，ある数式の集合 Y があるとき，元 $y \in Y$ のスコアは式 (5) で容易に求まる．したがって，次のようなスコアに関して N ベストの数式 $(y_1, y_2, \cdots, y_N)$ は式 (1) および前項の (Step 2) に準じた最大値問題として求まる．しかし，$Y(s)$ の元数は膨大な数となることが多い．例えば，数式曖昧表記文字列 “abc”から変換し得る数式の総数は，8通りの3ノード×5通りの2エッジの組み合わせであり，$\dim(Y(\text{“abc”})) = 12800$（通り）となる．そこで N ベストの算出アルゴリズムでは，次のような効率化を図った．

1）字句解析における $K(s)$ をキー分解数 $\dim(\boldsymbol{k}_i)$, $i = 1,...,K$ の小さい（可能性の高い）順にソートする．

2）構造解析における $T(\boldsymbol{k})$ を求める際の繰り返し回数に上限を設定し，上限を超えたら計算を打ち切る．

3）$W(\boldsymbol{t})$ を求める際に，演算子の作用するオペランド数式も N ベストの算出アルゴリズムによってスコア上位の組み合わせしか実施しない．

4 評価

本研究で設計した候補算出プログラムの予測精度を評価する．

4.1 評価用データ

今回の評価実験のために800個の数式サンプルデータを作成し，100個ずつ8組のデータセットを準備して，8分割の交差検定を行った．すなわち，1組（100個）を評価データとして，残り7組を訓練データに使用する．

本論文では，アルゴリズムの有効性を確かめるため，扱う数式の数学的範囲を高校の教科書「数学 I」[10] の「数と式」,「方程式と不等式」,「2次関数」に限定し，それらの解説ページおよびその章末問題に印刷されている数式を採用した．

また，数式曖昧表記文字列が長い数式の場合，候補算出時間がかかりすぎるので，文字数が16文字以内で数式が重複しないものを800個選んで，作成した．実験のためのデータセットを D とする．

$$D = \{(s_i, y_i) | (i = 0, ..., 799)\} \quad (10)$$

ここで，y_i が教科書に印刷されていた正解数式で，s_i はその数式曖昧表記文字列を表す．実際には独自の内部表現によってファイルに記録している．添え字 i は，ほぼ教科書に登場した順番に記録した．したがって，近い番号の数式データは数学の扱う分野も近い（同種の）可能性が高くなっている．そこで，データの8分割 D_t $(t = 0, ..., 7)$ には，できるだけ分散されるように式 (11) で定義した．

$$D_t = \{(s_i, y_i) \in D | i \bmod 8 = t, (i = 0, ..., 799)\} \quad (11)$$

4.2 評価方法

評価指標としては正解率を使う．

実験環境は MacOS 10.9, 3.2GHz Intel core i3, 8GB の PC を使い，3章で設計した Java 言語による訓練プログラムによって，20個ずつの訓練を進めたスコアパラメタを記録しながら実施した．

評価実験では，次に示す3つの条件に分けて実施した．

実験1 (Step 1) スコアパラメタ初期値：$\boldsymbol{\theta} = \mathbf{0}$
(Step 3) 式 (6) の通り．
$t = 0, ..., 7$ それぞれに対し，D_t を評価データ，$D - D_t$ を訓練データに使用．

実験2 (Step 1) スコアパラメタ初期値を実験1の700データ訓練後の値：$\boldsymbol{\theta} = \boldsymbol{\theta}_{700}$
(Step 3) 式 (6) の通り．
$t = 0, ..., 7$ それぞれに対し，D_t を評価データ，$D - D_t$ を再び同じ順番で訓練データに使用．

実験3 (Step 1) スコアパラメタ初期値：$\boldsymbol{\theta} = \mathbf{0}$
(Step 3) 次式 (12) で実施．

$$\begin{aligned} \theta_f &:= \theta_f + 2 \quad \text{for} \quad \{f \le F_{total} | x_f(y_i) = 1\} \\ \theta_{\bar{f}} &:= \theta_{\bar{f}} - 1 \quad \text{for} \quad \{\bar{f} \le F_{total} | x_{\bar{f}}(y_p) = 1\} \end{aligned} \quad (12)$$

$t = 0, ..., 7$ それぞれに対し，D_t を評価データ，$D - D_t$ を訓練データに使用．

5 結果と考察

5.1 結果

実験 1 の正解率の結果を表 2 に示す．Best1 が第 1 候補の正解率で，Best3 が第 3 候補以内に正解を含む正解率，Best10 が 10 ベスト以内に正解を含む正解率を表す．例えば，訓練回数 100 の結果は，訓練回数 20,40,60,80,100 のそれぞれの訓練後の評価データ $D_t(t = 0, \cdots, 7)$ に対する正解率の累積平均値である．それら正解率のグラフを図 3 に示す．また，訓練回数に対する式 (5) のスコア平均の推移を図 4 に示す．

表 2 実験 1 の精度（平均正解率）

訓練回数	Best1（%）	(SD)	Best3（%）	(SD)	Best10（%）	(SD)
0	25.9	3.8	41.3	4.4	52.3	4.3
100	62.7	14.7	75.5	9.2	81.5	6.8
200	75.6	6.6	82.7	5.0	86.6	4.3
300	79.3	4.1	85.2	4.3	88.1	4.3
400	79.2	3.8	85.1	3.8	88.2	3.5
500	80.0	4.4	86.7	4.0	89.5	3.3
600	79.5	3.7	85.9	3.4	89.2	3.8
700	79.1	5.7	85.7	5.3	89.2	4.2

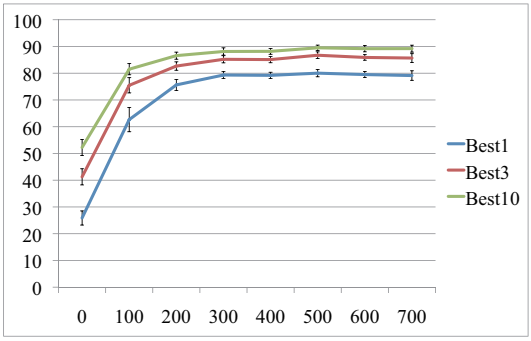

図 3 実験 1 の訓練回数-平均正解率

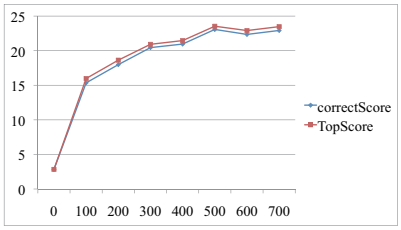

図 4 実験 1 の訓練回数-評価式平均スコアの推移

次に，実験 2 の平均正解率の結果を表 3 に示す．ただし，図 5 は訓練回数 800〜1400 が実験 2 のグラフであり，比較のため実験 1 のデータ（訓練回数 0〜700）も表示した．なお，訓練回数 800〜1400 と訓練回数 100〜700 がそれぞれ同じデータを 2 度使用して訓練を実施した．

実験 3 の平均正解率の結果は表 4 のようになった．それら正解率のグラフを図 6 に示す．また，訓練回数に対する式 (5) のスコア平均の推移を図 7 に示す．訓練回数に比例してスコア値が上昇している．

表 3 実験 2 の精度（平均正解率）

訓練回数	Best1（%）	(SD)	Best3（%）	(SD)	Best10（%）	(SD)
800	84.2	3.6	88.6	3.2	91.5	3.1
900	84.7	3.0	89.1	3.8	91.5	3.6
1000	83.4	3.8	88.4	3.4	90.6	3.7
1100	82.8	4.8	87.3	3.4	90.5	3.2
1200	82.1	3.8	87.8	2.8	90.3	2.4
1300	81.7	3.3	87.6	2.9	91.0	2.8
1400	82.3	3.4	88.0	2.2	91.4	2.4

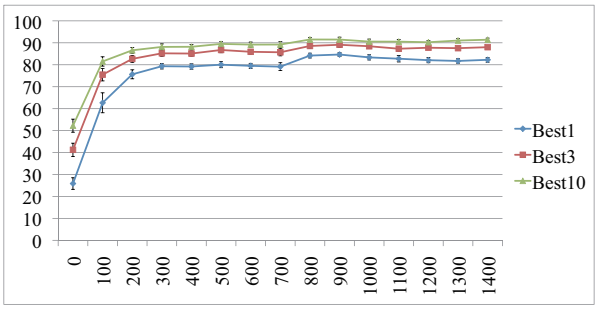

図 5 実験 2 の訓練回数-平均正解率

5.2 考察

5.2.1 実験 1 について

第 700 回目のベスト 1 の精度が 79.1%，ベスト 3 までの正解率が 85.7%，ベスト 10 までの正解率 89.2%と素朴なアルゴリズムにしては検討している．第 300 回の訓練でほぼこの正解率に達しており，第 500 回目以降は完全に横ばいである．

スコアの推移（図 4）見ると，第 500 回目以降はスコア 23 当たりを上下して不安定な変化である．実際，評価データ D の中では要素キー “a” に対して a の場合と α の場合が混在しており，それらのスコア $\theta_{(\text{“a”},a,V)}$ と $\theta_{(\text{“a”},\alpha,V)}$ の値が訓練回数に応じて+1 と-1 を交互に逆転して推移していた．

5.2.2 実験 2 について

実験 2 は同じ訓練データで 2 度訓練した場合の結果である．もちろん評価データ D_t を正解率測定に使う場合は，それ以外のデータのみを訓練に使用している．訓練データ 2 巡目の訓練回数 800〜1400 では，正解率にベスト 1,3,10 いずれも僅かな上昇が見られ，第 1400 回目のベスト 1 の精度が 82.3%，ベスト 3 までの正解率が 88.0%，ベスト 10 までの正解率 91.4% であった．第 700 回目と第 1400 回目の平均正答率の差の検定を行ったところ，ベスト 1,3,10 いずれも有意な差が見られた（ベスト 1 同士:$t(39) = 2.97, p < .05$, ベスト 3 同士:$t(39) = 2.58, p < .05$, ベスト 10 同士:$t(39) = 2.87, p < .05$）．

5.2.3 実験 3 について

実験 1，2 の訓練アルゴリズムでは式 (6) を採用しており，正解要素のスコアには 1 加算すると同時に，不正解要素のスコアを 1 減算するため 1 度は正解要素に登場したことがあっても訓練データによって，全く訓練に登場しない要素のスコア 0 より低くなってしまうことが

表 4 実験 3 の精度（平均正解率）

訓練回数	Best1（%）	(SD)	Best3（%）	(SD)	Best10（%）	(SD)
0	25.9	3.8	41.3	4.4	52.3	4.3
100	53.3	14.6	82.5	6.4	88.5	4.3
200	60.3	5.0	86.1	4.2	91.7	3.2
300	64.1	5.1	89.1	3.2	93.8	2.9
400	67.7	5.7	90.1	3.1	94.1	3.1
500	67.6	5.7	90.6	2.9	94.5	2.8
600	69.1	4.6	90.8	2.7	94.3	2.5
700	68.5	6.0	91.1	2.5	95.0	2.5

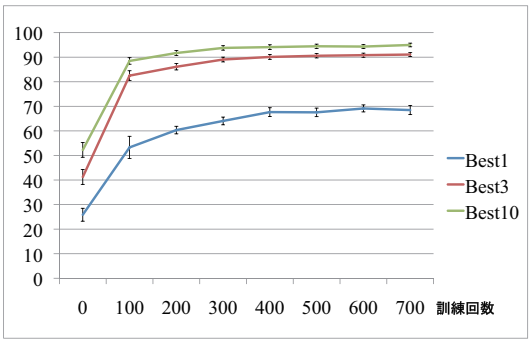

図 6 実験 3 の訓練回数-平均正解率

多く見られた．そこで訓練に式 (12) を採用したのが実験 3 である．第 700 回目のベスト 1 の精度が 68.5%，ベスト 3 までの正解率が 91.1%，ベスト 10 までの正解率 95.0%と，ベスト 1 の正答率はそれほど高くないが，ベスト 10 の正答率が高精度を得た．数式入力インタフェースに組み込んで利用する際には，ベスト 10 までの精度がかなり保証されていると実用性は高い．しかし，図 7 の様に，訓練回数の上昇と共にスコアも上昇し続けるため，対応が必要である．

5.2.4 計算時間と不正解事例

今回の実験では，N ベスト候補を算出する計算の前後の計算機内部時計の時刻を取得した差を計測した．それゆえ，正確な CPU 時間の測定にはなっていないことに注意してほしい．測定の結果，実験 1 の第 700 回目の平均計算時間は 623msec(SD=819) であり，答えを出すまでに時間がかかっている．しかも，数式によってかなりのばらつきがあり，長い式の場合は数分かかる場合もあった．アルゴリズムを改良し，計算時間を短縮することは今後の課題である．

また，今回のアルゴリズムでは計算打ち切りによるベスト候補探索漏れ（正解式のスコアの方が算出候補式のスコアより高い状態）を起こした場合は全体の 1%程度であった．この問題についても計算時間との兼ね合いがあり，検討課題である．

最後に，探索漏れやベスト 10 以内で正解できなかった数式の事例 $(s \to y)$ を示す．

`0.3.6.=4/11`	$\to$	$0.\dot{3}\dot{6} = \frac{4}{11}$
`D=42-4*(-1)*k`	$\to$	$D = 4^2 - 4 \cdot (-1) \cdot k$
`-2a2bc4`	$\to$	$-2a^2bc^4$
`x=-5+root13/2`	$\to$	$x = \frac{-5 \pm \sqrt{13}}{2}$
`<ABC=<DEF`	$\to$	$\triangle \mathrm{ABC} \equiv \triangle \mathrm{DEF}$

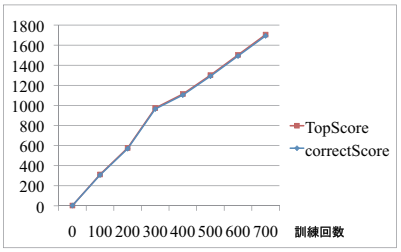

図 7 実験 3 の訓練回数-評価式平均スコアの推移

6 おわりに

本論文では，テキストベース数式入力（数式曖昧表記変換方式）に対して機械学習の最も素朴な方法で N ベスト候補算出アルゴリズムを提案した．800 データセットによる評価実験によって，正解率を調べ，実用レベルに近い精度を得た．しかし，評価のためのデータセットの数や種類（数学分野）を増やし，検証を続ける必要がある．今後は，高度な機械学習アルゴリズムへの適用を検討し，計算時間短縮と合わせて検討していきたい．

謝辞

本研究は JSPS 科研費（課題番号：26330413）の助成を受けたものです．

参考文献

[1] Pollanen, M., Wisniewski, T., Yu, X. : XPRESS: A Novice Interface for the Real-Time Communication of Mathematical Expressions, In Proceedings of MathUI2007,2007.

[2] 福井哲夫：数式のインテリジェントな線形入力方式，京都大学数理解析研究所講究録，Vol. 1780，pp. 160–171，2012.

[3] 福井哲夫：数式のインテリジェントな線形入力方式と評価，数式処理，Vol. 18，No. 2，pp. 47–50，2012.

[4] 福井哲夫：線形文字列変換による対話型数式入力方式の効果，京都大学数理解析研究所講究録，Vol. 1785，pp. 32–44，2012.

[5] 福井哲夫：対話型数式ユーザインタフェース MathTOUCH における数式表記表現の代数的ルールによる正準化の方法，京都大学数理解析研究所講究録「数学ソフトウェアと教育」，Vol. 1843，pp. 119–130，2013.

[6] 白井詩沙香，福井哲夫：数式自動採点システム STACK における数式入力方法の改善，コンピュータ&エデュケーション，Vol. 37，pp. 85–90 ，2014.

[7] 土方 嘉徳, 堀江 圭介，西田 正吾：N グラムモデルを用いた数式の予測入力インタフェースの開発，ARG Web インテリジェンスとインタラクション研究会，第 2 回研究会予稿集, pp. 33-38, 2013.

[8] 堀江 圭介，土方 嘉徳, 西田 正吾：N-gram モデルを用いた数式の予測入力アルゴリズムの提案，電子情報通信学会論文誌 D，Vol.J96-D, No.5, pp. 1255-1266, 2013.

[9] 徳永拓之：日本語入力を支える技術，株式会社技術評論社，2012.

[10] 飯高茂，松本幸夫，他 23 名：数学 I，高等学校数学科用文部科学省検定済教科書　2・東書・数 I 001，東京書籍株式会社，2005.