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

土方 嘉徳　　堀江 圭介　　西田 正吾
大阪大学大学院基礎工学研究科システム創成専攻
hijikata@sys.es.osaka-u.ac.jp horie@nishilab.sys.es.osaka-u.ac.jp nishida@sys.es.osaka-u.ac.jp

概要　数式は，理工学・社会・経済などあらゆる分野において自然現象・社会現象・工学技術等の知識を表現する最良の方法である．しかし，現在の数式の入力方式の煩雑さは，インターネットコンテンツにおける数式というメディアの普及を阻害している．そこで，本研究では入力された数式に対して次にユーザが行うであろう入力を予測する数式入力方式を提案する．自然言語処理の分野でよく用いられている確率的言語モデル 1 つである N-gram モデルを数式に適用する．数式は階層的な構造を持つので，その階層情報を利用した階層 N-gram モデルを提案する．このモデルに基づきユーザの入力を予測することで数式入力の煩雑さを軽減することを試みる．

キーワード　数式入力，予測入力，N-gram モデル

1　はじめに

数式は，理工学・社会学・経済学などあらゆる分野において自然現象・社会現象・工学技術等の知識を表現する最良の方法の 1 つである．World Wide Web Consortium（W3C）により，XML の拡張機能である Mathematical Markup Language（MathML）が公表され，Web 上での数式の表示が可能になった．また，MathML を用いることで数式の意味内容の記述も可能である．各種 Web ブラウザや数式処理ソフトも MathML に対応しつつある．それにともなって，数学の問題の解き方をお互いに教え合ったり，新たな解き方を公開したりするコミュニティサイト 1 も出現し始めている．また近年，数式をクエリとしたインターネット検索システムの研究もなされている [4, 8]．これらの事例から，Web 上で数式を記述することの潜在的なニーズは高まっていると考えられる．

しかし，インターネットにおいて，数式というメディアは依然として広く普及していない．その普及を阻む要因の一つとして，現在の数式の入力方式の煩雑さが考えられる．現在の一般的な数式入力方式は英語や日本語などの自然言語の入力方式と異なっている．自然言語はキーボードのみから入力できるのに対し，数式の入力はキーボードとマウスを併用して入力しなければならない．数式は単に数字，記号，アルファベットを一列に並べたものではなく，分数や指数などの構造を持って表現される．構造はキーボードからは入力できないため，数式エディタ内のボタンを用いて入力する．キーボードとマウスを併用しなければならない点が数式入力の煩わしさにつながっていると思われる．

そこで，本研究ではユーザがこれまでに入力した数式を用いて次にユーザが入力するであろう数学記号や文字を予測する入力方式を提案する．具体的にはキーボードから入力できない構造や文字の入力を予測し提示する．ユーザはキーボードを用いて提示された予測候補を選択する．

2　関連研究

2.1　数式入力

現在，数式を記述する方法の 1 つとして TeX がよく用いられる．しかし，TeX などのマークアップ言語で数式を記述する際，ユーザは自分の入力した数式を視覚的に確認しながら入力することはできない．そこで，ユーザが自分の入力した数式を視覚的に確認しながら数式を記述するツールとして，数式エディタが存在する．実用化されている数式エディタとしては Microsoft Office の数式エディタやその機能拡張版である MathType および InftyEditor が挙げられる．MathType や InftyEditor では，マウスとキーボードを併用する煩わしさを取り除くために構造的な部分をコマンドで入力する手法を採用している．しかし，数式入力に不慣れなユーザにとって，コマンドを習得するためには努力が必要であると思われる．

また，数式入力に関する研究としては手書き数式入力 (手書き数式認識) の研究が数多く存在する [7, 9]．これらの手書き数式入力では，ジェスチャ（ユーザの入力ストローク）を用いて記号や文字の分割を行なっている．分割した文字や記号はデータベース中の文字や記号とのマッチングを行い文字認識を行う．さらに，入力された手書き数式を構造識別器に通して，各文字（または記号）間の構造的な識別を行う．これらのプロセスにより，手書きで入力された数式を認識し MathML や TeX などで表された数式を出力する．

これらの手書き数式入力はタブレット PC 等のペンベースのデバイスに対しては非常に有用であり，既存の数式エディタの煩わしさを取り除く効果を持つ．しかし，

一般的なPCで数式を手書きで記述することは直観的でなく，ユーザに負担を伴うと考えられる．我々の予測入力方式は特別なハードウェア環境を必要とせず一般的なPCで利用できる．また，我々が調査した範囲では，数式に対する予測入力を扱う論文は発見できなかった．

2.2 自然言語処理分野における予測

自然言語を対象とした予測入力に関する研究は1980年代から存在する．これら自然言語を対象とした予測入力では，数式を対象とする入力と異なる点が存在する．自然言語は一般的に単語単位で予測される [3]．そして，予測される単語はその単語を打ち始めた途中までの情報を用いて予測されることが多い [1]．しかし，数式ではある文字や記号が入力された際，次に入力される文字や記号は非常に多様になると考えられる．よって，数式にこのような辞書を用いた手法を適用することは困難であると言える．そこで，我々は確率的な言語モデルの1つであるN-gramモデルを用いて数式を予測する（N-gramモデルについての詳細については3章で説明）．N-gramモデルを用いて予測を行うためには，各文字や各記号の出現頻度の情報のみを保持すればよく，数式の予測にも適用できる．

N-gramモデルを用いた自然言語の予測の代表的な研究がReactive Keyboard[2] である．彼らの研究においてはアルファベット一文字を木構造のノードとし文字列の木構造を構築する．N-gramの出現回数により各ノードに優先順位を付ける．優先順位に従いユーザが行った入力と木構造内のノードの照合を行い，適合箇所の子ノードが予測候補となる．我々の手法でもN-gramモデルを用いて予測を行う．しかし，我々は数式の階層情報を含んだモデルを用いており，彼らの手法よりも数式に適した手法となっている．

3 予測手法の設計

3.1 数式の階層

まず，以下の定式化で必要な用語の定義を与える．我々は，キーボードから入力される数字やアルファベット，()や=などの記号を「文字」と呼ぶ．また，キーボードから入力できない文字や記号[1]も便宜的に「文字」と呼ぶこととする．上付き，下付きを入力した際には，表現的には文字や記号は生成されないが，仮想的に「上付き」,「下付き」という1つの文字が入力されたものとみなす．またこれらの文字のうち，大型演算子，分数，上付き，下付き，アクセントを入力した場合，追加的な入力を行う必要がある．例えば，分数を入力した際には分数の項が生成され，さらに分母，分子を入力する必要がある．同様に，インテグラルを入力した際には積分項が生成され，追加的に積分範囲を入力する必要がある．このとき，分数の項の中には分母，分子，積分項の中には積分範囲が包含されている関係にあると言える．このように1つの入力によって生成される項の中にさらなる入力が必要な要素（箇所）を包含している文字が存在する．我々は，この包含関係を「構造」と定義する．さらに，包含関係にない文字同士は同一の「階層」に存在し，ある文字を入力した際に生成される項に包含される追加的な入力内容は，ある文字が存在する階層よりも1つ深い「階層」に存在すると定義する．

3.2 階層 N-gram モデル

我々は，ユーザが入力した数式のログデータを取得した後に，そのログデータをN-gramモデルと呼ばれるモデルでモデル化する．なお，$N = 1, 2, 3$ の場合をそれぞれユニグラム（unigram），バイグラム（bigram），トライグラム（trigram）と呼ぶ．ここで，N-gramモデルの確率値は

$$P(w_n|w_{n-N+1}^{n-1}) = \frac{C(w_{n-N+1}^{n})}{C(w_{n-N+1}^{n-1})} \tag{1}$$

で与えられる．この確率値を $N-gram$ 確率と呼ぶこととする．我々はこの確率値を数式の各文字に対して計算し，それを用いて予測を実現する．ここで $C(w_1^n)$ は単語列 w_1^n が学習データ中に出現する回数である．

このモデルを用いて数式をモデル化する際には，問題点が存在する．数式は表現上[2]，構造を持つことがありシーケンシャルな記号列ではない．そこで，我々は数式の構造の情報をモデルに組み込むためにユーザの数式入力のログ（以降，数式入力ログデータ）を数式の階層別に分割してモデルを構築することを提案する．N-gramモデルに直接，階層構造の情報を組み込むことはできないので，階層別に複数のモデルを持っておくことで数式が階層的な構造を持つ問題に対処する．つまり，数式入力ログデータを階層毎に分割し，分割後のログデータそれぞれに対してN-gram確率（式1）を計算する．予測を行う際には，ユーザが入力している階層に対応するN-gram確率に基づいて予測を行う．

3.3 N-gram モデルのスムージング

ただし，N-gramモデルにおいてはゼロ頻度問題という問題が存在する [5]．これは、出現頻度によりN-gram確率を推定すると，学習用データセット中に出現しない単語組の確率値を0としてしまうという問題である．これを解決するために，線形補間法 [5] というスムージング手法を用いる．線形補間法は，N-gram確率 $P(w_n|w_{n-N+1}^{n-1})$

[1] キーボードから入力できない文字とは，大型演算子（総和演算子，積分演算子など），分数，上付き，下付き，数学記号（微分記号（∂）や説明用記号（$\exists$,$\forall$）など），演算子，アクセント（ˆ，˜ など）を指す．

[2] 表現上とは，自然言語の外見上の並びのみに着目した場合を指す．

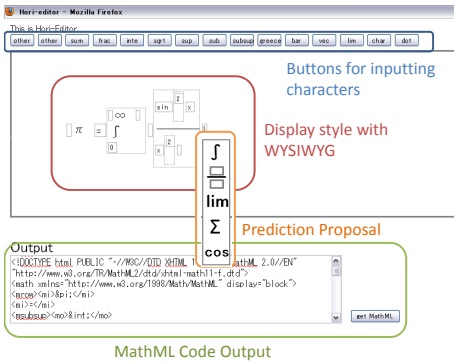

図 1 数式予測入力インタフェース

を低次の M-gram($M < N$) の確率値と線形に補間する方法である．$N = 2$（バイグラム）の場合は，N-gram 確率は次のようになる．

$$P(w_n|w_{n-1}) = \lambda P(w_n|w_{n-1}) + (1-\lambda)P(w_n) \quad (2)$$

ここで，λ は補間係数であり，経験的に $\lambda = 0.7$ と設定する．また，$N = 3$（トライグラム）の場合には，N-gram 確率を次のように書きなおすことができる．

$$\begin{aligned} P(w_n|w_{n-2}w_{n-1}) &= \lambda_3 P(w_n|w_{n-2}w_{n-1}) \\ &\quad + \lambda_2 P(w_n|w_{n-1}) \\ &\quad + \lambda_1 P(w_n) \end{aligned} \quad (3)$$

ここで，$\lambda_3, \lambda_2, \lambda_1$ は，それぞれトライグラム，バイグラム，ユニグラムに対する補間係数であり，$\sum_i \lambda_i = 1$ となるように設定される．これらの補間係数に関しては，経験的に $\lambda_3 = 0.5$, $\lambda_2 = 0.3$, $\lambda_1 = 0.2$ と設定する．以降，スムージング手法についてはこれらの補間係数を用いることとする．

4 数式入力インタフェース

3 章で提案した予測手法を実装したインタフェースを作成した．このインタフェースは JavaScript で実装を行なっており，Web ブラウザ上で動作する．図 1 に我々の実装したインタフェースを示す．我々の実装したインタフェースは，一般的な数式入力エディタに準拠している．すなわち，WYSIWYG による入力インタフェースと，キーボードから入力できない文字を入力するためのボタンを備えている．

次に，予測結果の提示と選択について述べる．ユーザが入力する位置にカーソルを移動させれば，そのカーソルの下に，数式の予測候補を列挙する．ここでのカーソルの移動はカーソルキーで行う．予測候補を選択するには，スペースキーなどでフォーカスを予測結果表示欄に移動させて，さらにスペースキーやカーソルキーで入力したい予測候補までフォーカスを移動させ，エンターキーで確定する．また，[4] などの数式検索システムのクエリを生成するために，入力された数式の MathML コードを出力する．

5 予測精度の評価

本章では，数式入力において提案する予測手法の評価を行う．本評価の目的はいくつかのベースライン手法と比較したときに，提案手法は優れた予測結果を出力することができるかついて明らかにすることである．

5.1 評価用データと予測精度の算出方法

評価用の数式として，解析学の教科書 [6] 中の定理と問題から 1000 個の数式を手動で取得する．6 名の協力者を 3 人ずつの 2 グループに分割し，各グループのメンバーにそれぞれ 500 個の数式を入力してもらう．合計で 1000 個の数式に関して 3000 個の数式入力ログデータセットを得る．この数式入力ログデータを同じ数式に対するログデータは同じグループに入るように 10 分割し，9 つを学習用データセット，残り 1 つを評価用データセットとする．予測精度の算出方法は，評価用データセット中の 100 個の数式をそれぞれ先頭から順に入力していくことを想定し，各入力位置において予測を行う．予測候補の中の何番目の候補がログデータの入力内容と一致するかを調査する．100 個の数式の各入力位置において，予測候補中の上位 k 位までに正解が出現する割合を予測精度とする．

5.2 提案手法とベースラインの予測精度の比較

5.2.1 提案手法とベースライン手法

提案手法についてはモデルの実装可能性を考慮し，$N = 3$ までについて調査する．$N = 4$ 以上の N-gram モデルを構築すると，N-gram 確率の数が膨大になり多くのメモリが必要となり実装が困難になる．提案手法（階層 N-gram モデル）の N=2, 3 の場合とそれぞれのスムージング手法について予測精度を調査する．

比較手法として以下の 4 つの手法を挙げる．

- **直近予測**：k 個前の入力内容を k 位の予測候補とする．ただし、$k = 10$ までとし，$k < 10$ の時，つまり直前の入力が 10 個未満の時は予測候補は k 位までを出力する．
- **直近頻度順予測**：上述の直近予測の予測候補を出現頻度順に並び替える．出現頻度は入力中の 1 つの数式の中に出現した回数をカウントすることで得る．頻度が同じ場合は、直近予測の候補順に準ずる．
- **ユニグラムモデル（$N = 1$）**：N-gram モデルの $N = 1$ の場合である．予測候補は学習用データ

セット中に出現する頻度が高い順となり，常に同じ候補が表示される．

- **ユニグラムモデル（$N = 1$）＋直近予測**：直近予測における N 位が正解である確率値を事前に学習用データセットから計算しておく（表 1 中の直近予測の確率値）．この確率値を N 個前に入力された文字のスコアとして、ユニグラムモデルから出力される確率値に足し合わせる．足しあわされたスコアの順に予測候補を提示する．

5.2.2 予測精度の結果

各評価用データセット中の数式を対象として，予測精度を算出した結果を表 1 に示す（表中の太字は，最も精度が高いものを表している）．結果よりすべての手法のうち，提案手法の $N = 3$（スムージング）が最も高い予測精度を示していることがわかる．インタフェース上に表示する候補を 5 位以内に限る場合に正解が出現する確率は約 89%，10 位以内では約 95%の予測精度を実現している．

6 ユーザビリティの評価

6.1 評価の目的

5 章では，予測精度によって我々の提案手法がその他の予測手法よりも優れていることを示した．しかし，予測精度の評価のみでは我々の提案インタフェースがユーザの数式入力を支援できているかどうかわからない．本章の目的は，我々の数式入力インタフェースのユーザビリティに関する評価を通して，我々の提案したインタフェースが実際にユーザの数式入力を支援できていることを示すことである．

6.2 評価実験設定

我々の提案インタフェースのユーザビリティを評価するにあたって，評価指標は定量的なものと定性的なものを用いる．定量的な評価指標として，マウスとキーボードの切り替え回数，入力時間，誤入力回数を用いる．定性的な評価指標として，インタフェースのユーザビリティに関するアンケート（多肢選択式の質問項目と自由記述）を用いた．被験者は大学生・大学院生 12 名である．事前アンケートにより彼らにコンピュータ上での数式の入力経験を尋ねた．全ての被験者は，コンピュータ上での数式入力経験があり，一般的な数式エディタもしくは TeX を用いた数式入力の経験があると答えた．被験者のうち 3 名は，TeX による数式入力に長けており大部分の数式を TeX コマンドのリファレンスを見ることなしに入力できる．

6.3 インタフェースの違いに関する評価

この節では，数式を予測入力を用いて入力するという我々の手法の有効性を示す．そのために，我々の提案する数式入力インタフェースと予測を用いない一般的な数式エディタ，TeX を用いる WYSIWYG でない数式入力インタフェース，TeX を用いる WYSIWYG な数式入力インタフェースの比較を行う．

6.3.1 実験方法

本実験で用いる我々の提案インタフェースは 4 章で説明したものである．一般的な数式入力エディタ（以下，予測なしインタフェース）は，我々の提案インタフェースから予測入力機能を除いたものである．TeX を用いる数式入力インタフェースは，我々が JavaScript を用いて独自に実装を行ったもの（以下，TeX インタフェース）と数式エディタである MathType[3]，文書作成ソフト LyX[4]の中に含まれる数式入力機能（以下，LyX）を使用する．

我々が実装を行った TeX インタフェースでは，テキストエリアに TeX コマンドを用いて数式を入力し，コンパイルボタンをクリックすると TeX コマンドによって入力された数式を視覚的に確認することが出来る．この TeX インタフェースでは，コマンドから変換した数式に対して直接編集を行うことはできない．つまり，このインタフェースは WYSIWYG な表示形式を実現していない．

一方，MathType と LyX は WYSIWYG な表示形式を実現している．これら 2 つのインタフェースでは，ユーザは TeX コマンドの一部を入力した後にスペースキーもしくはエンターキーを押すことで入力した TeX コマンドを数式の形に変換できる．また，変換後の数式に対して直接編集を行うこともできる．これら 2 つのインタフェースの大きな差異は，TeX コマンドの補完機能である．LyX では，ユーザがコマンドの一部を入力すると残りの部分を推定して補完する機能が存在する．一方，MathType ではこのような機能は存在しない．また，入力中に TeX コマンドがわからない場合は TeX コマンドのリファレンスを見ても良いこととする．

各インタフェースを用いてそれぞれ 20 個，計 100 個の数式を入力してもらい，各数式について定量的指標を測定した．これらの数式はすべて異なるものを用いるが，各数式の入力量は一定の範囲内に収まるようにした．また，これらの数式は解析学の問題集から収集し，学習用データセット中の数式と同一のものが入らないように注意した．なお我々の提案インタフェースの学習用データセットには，5.1 節で用いた 3000 個の数式入力ロ

[3] http://www.dessci.com/en/products/mathtype/
[4] http://www.lyx.org/

表 1 上位 k 位以内に正解候補が出現する割合

	k = 1	2	3	4	5	6	7	8	9	10
直近予測	0.154	0.286	0.353	0.393	0.410	0.414	0.416	0.416	0.416	0.416
直近頻度順予測	0.181	0.295	0.359	0.394	0.411	0.415	0.416	0.416	0.416	0.416
N=1	0.287	0.496	0.571	0.595	0.638	0.681	0.702	0.728	0.758	0.793
N=1 +直近予測	0.242	0.498	0.614	0.668	0.710	0.736	0.787	0.810	0.826	0.844
N=2	0.546	0.703	0.778	0.826	0.856	0.880	0.898	0.912	0.925	0.935
N=3	0.627	0.730	0.780	0.806	0.819	0.826	0.833	0.839	0.843	0.845
N=2（スムージング）	0.541	0.689	0.758	0.798	0.838	0.873	0.899	0.916	0.930	0.940
N=3（スムージング）	**0.657**	**0.770**	**0.833**	**0.844**	**0.888**	**0.907**	**0.921**	**0.934**	**0.945**	**0.954**

表 2　インタフェースの違いに関する評価の定量的評価結果

	提案 IF	予測なし	TeX	LyX	MathType
マウス入力回数	1.21	13.2	0	0	0
キーボード入力回数	90.0	52.2	121	86.0	98.1
切り替え回数	1.26	16.5	0	0	0
入力時間 [s]	53.7	74.0	89.6	64.1	67.8
誤入力回数	1.55	1.57	4.43	5.81	6.08

表 3　インタフェースの違いに関する評価の定量的評価に対する t 検定の結果

	切り替え回数	入力時間	誤入力回数
提案 IF vs. 予測なし	*	*	
提案 IF vs. TeX	*	*	*
提案 IF vs. LyX	*	*	*
提案 IF vs. MathType	*	*	*
予測なし vs. TeX	*	*	*
予測なし vs. LyX	*	*	*
予測なし vs. MathType	*		*
TeX vs. LyX		*	
TeX vs. MathType		*	
LyX vs. MathType			

$*: p < 0.05$

グデータを全て使用した．100 個の数式を入力し終えた後に，定性的な評価指標に関するアンケートに回答してもらった．アンケートの質問項目を表 4 に示す．どのインタフェースが最も快適に入力できたか，また直観的に入力できたかという観点からそれぞれ Q1.，Q3. を尋ねた．実際の入力時間と体感の入力時間が異なる可能性を考慮して Q2. を尋ねた．また，インタフェースに慣れるまでの早さを測るために Q4. を尋ねた．総合的にどのインタフェースを最も嗜好するかを Q5. で尋ねた．

6.3.2　定量的評価項目の結果

インタフェースの違いに関する評価の定量的な評価指標の結果を表 2 に示す．表 2 では参考までに，キーボードのキー押下回数とマウスのクリック回数も示す．これらの指標は全て 1 数式当たりで計算する．また，これらの結果の間に統計的に有意な差が見られるかを t 検定を用いて検定した．その結果を表 3 に示す．表 2，3 中の “提案 IF” は我々の提案インタフェース，“予測なし” は予測なしインタフェース，“TeX” は我々の実装した TeX インタフェースを表している．

表 2 の入力時間の項目より，提案手法が最も短い時間で入力できていることがわかる．入力時間はスムーズに数式が入力できていることを表す重要な指標であると考えられる．また提案インタフェースと予測なしインタフェースを比較すると，我々の提案インタフェースが数式入力の煩わしさの原因の 1 つであると考えられるキーボードとマウスの切り替え回数を大きく削減できていることがわかる．この切り替え回数の削減率はおよそ 89.1%であり，予測入力を用いることで数式入力の煩わしさを大きく軽減できたといえる．

また，誤入力回数については，我々の提案インタフェースと予測なしインタフェースに比べて 3 つの TeX を用いる数式入力インタフェースがおよそ 3 倍から 4 倍多くなった．この結果からコマンドベースでの数式入力では誤入力回数が多くなるという傾向があることが観察される．コマンドベースの数式入力ではユーザはコマンドから数式への変換を要求されるが，我々の提案インタフェースおよび予測なしインタフェースでは直接入力したい記号を選択できる．変換時にはユーザの意図と異なる結果が表示されたり，コマンドを入力した時点では気づかなかったエラーを発見することがある．この特性により我々の提案インタフェースおよび予測なしインタフェースで誤入力回数が減少したと考えられる．また，予測入力の有り無しは誤入力回数には大きな影響を及ぼさない．

6.3.3　アンケートの結果

インタフェースの違いに関する評価のアンケート結果を表 4 に示す．各項目の数字は，アンケートでその項目を選択した被験者の数である．() 内の数字は, 事前アンケートで TeX を用いて何も見ずにほとんどの数式を入力できると回答したユーザ（以下，TeX ユーザ）の数である．Q1. から Q5. までの全ての項目において，我々の提案インタフェースが最も良い評価を得ていることが見て取れる．特に，Q3. に関してはすべての被験者から我々の提案インタフェースが最も直観的に入力できたとの回答を得た．我々の提案インタフェースと異なり，コマンドベースの手法ではユーザはコマンドを入力した

表 4 インタフェースの違いに関する評価のアンケート結果

	提案 IF	予測なし	TeX	LyX	MathType
Q1. どのインタフェースが最もスムーズに入力できたか	8 (1)	0	0	1	3 (2)
Q2. どのインタフェースが最も短い時間で入力できたか	11 (2)	0	0	0	1(1)
Q3. どのインタフェースが最も直観的に入力できたか	12(3)	0	0	0	0
Q4. どのインタフェースが最も早く慣れることができたか	8(1)	0	0	0	4(2)
Q5. どのインタフェースを最も好むか	8(1)	0	0	0	4(2)

後，変換を行なって入力結果を確認しなければならない．一方，我々の提案手法では入力したい文字を直接選択できる点が直観的であったと考えられる．予測なしインタフェースでもユーザはコマンドの変換を必要としない．しかし，予測なしインタフェースではユーザがキーボードからマウスに切り替える回数が多い．2 つの入力デバイスを切り替えることの煩わしさから，直観的な入力の観点での予測なしインタフェースの評価が低くなったと考えられる．

最もスムーズに入力できたインタフェースの項目でも我々の提案インタフェースが最も多くの支持を得た．自分の入力したい文字が予測候補の上位に出現するケースが多いことが，ユーザがスムーズに入力できる要因の 1 つであると考えられる．TeX を用いる 3 つのインタフェースでは誤入力回数が多かったため，この観点の評価が低下したと考えられる．しかし，一部の被験者は TeX を用いるインタフェースである LyX と MathType を支持した．これは，LyX と MathType のマウスを全く用いることなくキーボードのみから入力できる点を評価したものであると考えられる．被験者の体感時間に対する項目では，定量的な評価項目と一致する結果が得られた．インタフェースに対する慣れの早さでも我々のインタフェースが最も多くの良い評価を得たが，MathType も一部の被験者の支持を得た．この点に関しては，MathType は主に TeX による数式入力の経験が多い被験者から支持された．

最後にアンケートの自由記述の結果について考察する．アンケートから，“マウスとキーボードを切り替えることがユーザの大きな負担になる” という意見が多く得られた．これは，マウスとキーボードの切り替え回数を減少させることで数式入力の煩わしさを軽減させるという我々の提案インタフェースを支持する結果であるといえる．

7 おわりに

本研究では入力された数式に対して次にユーザが行うであろう入力を予測する数式入力方式を提案した．自然言語処理の分野でよく用いられている確率的言語モデルの中から N-gram モデルを数式に適用した．数式のもつ構造の情報をモデルに組み込んだ階層 N-gram モデルを提案し，そのモデルから出力される確率を予測入力に用いた．また，評価用データを取得し，予測精度によって我々の手法の有効性を示した．また，被験者を用いたユーザビリティ評価実験を行った．我々の提案インタフェースが予測なしインタフェースや TeX による数式入力インタフェースよりもユーザビリティが高いことを定量的評価項目とアンケートを用いて確認した．

我々の研究の今後の課題として，複数の入力の組み合わせを予測することが挙げられる．現在のインタフェースでは，ユーザの 1 入力を予測対象としている．もし，ユーザの複数入力を高い精度で予測することが可能になれば，ユーザの入力量を大きく減少させることが出来る．複数入力の予測を行う具体的な手法については，今後検討していくべき課題である．

参考文献

[1] A. Copestake, “Augmented and Alternative NLP Techniques for Augmented and Alternative Communication,” Proceedings of Natural Language Processing for Communicaton Aids Workshop, pp.37–42, 1997.

[2] J. Darragh, I. Witten, M. James, “The Reactive Keyboard: A Predictive Typing Aid, ” Computer, Vol.23, No.11, pp.41–49, 1990.

[3] N. Garay-Vitoria and J. Abascal, “Text prediction systems: a survey,” Universal Access in the Information Society, Vol.4, No.3, pp.188–203, 2006.

[4] H. Hashimoto, Y. Hijikata, and S. Nishida, “Incorporating Breadth First Search for Indexing MathML Objects,” IEEE International Conference on Systems, Man and Cybernetics, pp.3519–3523, 2008.

[5] 北 研二，言語と計算 確率的言語モデル，東京大学出版，東京，1999.

[6] 三宅 敏恒，入門微分積分，培風館，東京，1992.

[7] T. O’Connell, C. Li, T. Miller, R. Zeleznik, J. LaViola Jr., “A Usability Evaluation of AlgoSketch: A Page-based Application for Mathematics, ” Proceedings of SBIM, pp.149–157, 2009.

[8] R. Zanibbi, D. Blostein, “Recognition and Retrieval of Mathematical Expressions, ” International Journal on Document Analysis and Recodnition, pp.1–27, 2011.

[9] R. Zeleznik, A. Bragdon, F. Adeputra, H. Ko, “Hands-On Math: A Page-based multi-touch and pen desktop for technical work and problem solving, ” Proceedings of UIST’10. ACM Press, pp.17–26, 2010.