ニューラルコンピューティングによる特徴抽出手法および最適化手法の確立

慶応義塾大学　環境情報学部　教授　武藤佳恭
（共同研究　斉藤孝之，岡　宗一）

ハイパースペクトラム画像による人物の肌の特徴抽出

概要

本技術開発では、不特定の人物を正確に検出するための画像認識システムのエンジンをニューラルネットワーク技術および人工網膜チップや赤外カメラ等のハード技術を駆使することにより実現を試みる。その基礎研究として、赤外線カメラの熱情報と可視光カメラの色情報から人物の肌を効果的に抽出する自己組織化技術を提案する。

1　はじめに

ハイパースペクトラム・カメラによって撮影されたマルチバンド画像は、通常のCCDカメラでは観察されにくい情報を広域に渡って含んでいる(図1)。リモートセンシングをはじめとするマルチバンド画像解析法の一つにクラスタリングがある。マルチバンド画像のクラスタリングとは、ある同一領域を異なるp種類のバンドで撮影してできるp枚の画像を基に、各ピクセルをp次元の特徴ベクトルの類似度によっていくつかのクラスタに分けることである。現在のところ知られている代表的なクラスタリングの手法としては、ウォード法やセントロイド法をはじめとする古典的な階層的クラスタリングや、コホーネン等の提唱する教師無しニューラルネットを用いた自己組織化モデルなどがある[1,2]。階層的クラスタリングは局所解に陥りやすい上、計算オーダが膨大であるという欠点を含んでいる。コホーネンの自己組織化モデルは階層的クラスタリングよりは計算オーダや局所解の問題点を解決しているが、改善の余地は多い。本研究では、教師無しニューラルネットを用いたコホーネンの自己組織化モデルと武藤等のマキシマムニューロンモデルを融合した新しい自己組織化モデルを提案し、従来の自己組織化モデルと実験に基づく比較検討を行った。

2　アルゴリズム

2.1　マキシマムニューロンによる手法

P次元の特徴を持つN個のピクセルをM個のクラスターに分類するとき、$M \times N$個のニューロンを用意する。ニューロンnmへの入力をU_{nm}、出力をV_{nm}と表し、ピクセルnがクラスターmに属しているならば$V_{nm}=1$,属していないならば$V_{nm}=0$としてクラスタリングの状態を表現する。s枚目の画像におけるピクセルkの濃度(0～255)をx_{sk}とすると、ピクセルkの特徴ベクトルX_k、クラスターlの中心の特徴ベクトル$\overline{X_l}$は次の式で与えられる。nlはクラスターlに属しているピクセルの数である。

$$X_k=(x_{1k},x_{2k},...,x_{Pk}),\ \overline{X_l}=(\sum_{k=1}^{N}X_kV_{kl})/nl \quad (1)$$

ピクセルkとクラスターlの距離を示す値R_{kl}を、ユークリッド距離の2乗に基づいて次のように定める。

$$R_{kl}=(X_k-\overline{X_l})^2 \quad (2)$$

目的関数Eは、各ピクセルをクラスターに分類したときに生じる誤差2乗和で与えられる。

$$E = \sum_{k=1}^{N} \sum_{l=1}^{M} R_{kl} V_{kl} \quad (3)$$

一般に、Eの値が小さいほど、良い画像のクラスタリング結果であるといえる。ニューロンのダイナミックスの式を次のように設定する。

$$dU_{kl}/dt = -R_{kl} \quad (4)$$

ニューロンの出力 Vは次のようなマキシマムニューロンのルールによって決定される。tはイタレーションステップである。

$$V_{km}(t+1) = 1 \ \ if \ U_{km} = max[U_{kl}(t); \forall l]$$
$$V_{km}(t+1) = 0 \ \ otherwise \quad (5)$$

計算手順は以下の通りである。

1 Uの値をランダム値で初期設定をする。

2 式(5)のルールに従って各々のニューロンの出力Vを決め、クラスタリングの状態を遷移させる。

3 各々のクラスターについてクラスターの中心の特徴ベクトルを計算する。

4 各々のニューロンについてRの値を式(2)に基づいて計算し、各ニューロンへの入力の変化量を式(4)に基づいて計算する。

5 各々のニューロンについて、以下の1次のオイラー法を用いてUの値を更新する。

$$U_{kl}(t+1) = U_{kl}(t) + dU_{kl}/dt \quad (6)$$

6 Eの値の変化が停止するまで、2～5を繰り返す。

2.2 コホーネンによる手法

P次元の特徴を持つN個のピクセルをM個のクラスターに分類するとき、p次元の重みベクトルWをW_1～W_MまでM個用意する$(W_l = (w_{1l}, w_{2l}, ..., w_{Pl}))$。$X_k$、$\overline{X_l}$の定義は式(1)と同様とする。計算手順は以下の通りである。

1 $W_l (l = 1, 2, ..., M)$の値ランダム値で初期設定する。

2 各々のピクセルに対して、以下の計算を行なう。

$$(X_k - W_l)^2 \ \ (l = 1, 2, ...M) \quad (7)$$

$$(X_k - W_z)^2 = min(X_k - W_l)^2 \quad (8)$$
$$(l = 1, 2, ...M)$$

ユークリッド距離の2乗に基づいてM個の重みベクトルとの差を計算し、式(8)を満たす重みベクトルW_zの値を、式(9)に基づいて修正する。この時点で、ピクセルkはクラスターzに属しているものとする。

$$W_z(t+1) = W_z(t) + a(X_k - W_z) \quad (9)$$
$$(0 < a < 1)$$

3 $W_l \approx \overline{X_l} (l = 1, 2, ..., m)$となるまで、2を繰り返す。(学習の初期には$a = 0.8$程度とし、学習が進むにつれて$a$の値を0に近付けていく。)

2.3 ハイブリッドモデル

1 最初に、マキシマムニューロンを用いたクラスタリング処理を行なう。

2 エネルギーEの値の減り具合が小さくなってきたら、マキシマムニューロンによる処理結果を受け継いで、コホーネンの手法によるクラスタリング処理を行なう。

3 シミュレーション結果

ハイパースペクトラム・カメラ(図2)によって撮影された可視光画像と赤外画像を画角調整し、前処理で生成された赤外、R、G、B、色相、彩度の6枚の画像データを入力する。本シミュレーションで用いた画像はサイズ128×96、256階調のグレイスケールである。シミュレーションで用いたデータとハイブリッドモデルによる実行結果を図1に示す。

4 考察

人間の肌が放出する熱は個人差が少なくほぼ一定であり、赤外カメラでその熱をとらえることで効果的な特徴抽出が可能になる。熱情報は明るさの変化等の外乱を受けにくいという利点もある。シミュレーションではクラスター数が7～10のとき人物の肌が鮮明に抽出された。マキシマムニューロンは離散的なモデルであるため、最初はダイナミックにエネルギーが減少するが、局所解に陥ると脱出が困難である。コホーネンによる手法は連続的なモデルであるため、局所的に精密な解を捜し出すことができるが、収束に時間がかかる上、初期状態に依存して局所解に陥ることがある。ハイブリッドモデルは二つの手法の長所を組み合わせて計算時間と解の質を改善している。ハイブリッドモデルの場合、どのタイミングで離散モデルから連続モデルへと移行させるかが重要であり、局所解の回避を決定づける要因になる。

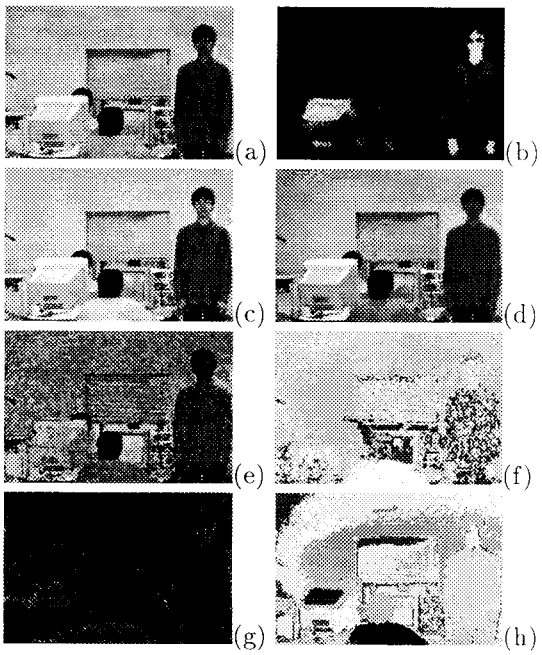

図 1: ハイパースペクトラムカメラによって撮影された (a) 可視光画像と (b) 赤外画像。可視光画像を (c)R、(d)G、(e)B、(f) 色相、(g) 彩度に分解し、(b)～(g) の 6 枚の画像を入力する。(h) 処理結果において人物の肌が特徴抽出されている。

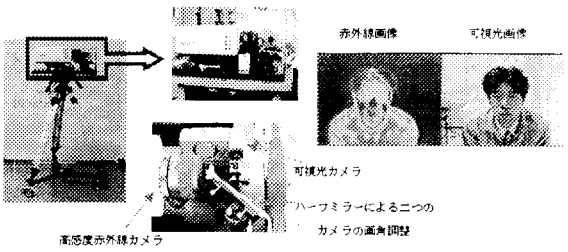

図 2: 可視光画像と赤外画像が同時撮影できるハイパースペクトラム・カメラ。

バンド	波長
X線	0.5A°～40A°
紫外線	50A°～0.3μm
可視光	0.4μm～0.7μm
赤外線	0.8μm～14.0μm
遠赤外線	14.0μm～1.0mm
マイクロ波	1.0mm～10.0mm

図 3: ハイパースペクトラムカメラの波長域。

参考文献

[1] Yoshiyasu Takefuji and Souichi Oka, "Innovations in Materials Design on the Information Highway: Electronic Prototyping - Classification, Discretization, and Functional Mapping", Proc. of Pacific forum on Intelligent Processing and Manufacturing of Materials, Australia, 1997.

[2] Souichi Oka, Tomoaki Ogawa, Takayoshi Oda and Yoshiyasu Takefuji, "A New Self-Organization Classification Algorithm for Remote-Sensing Images", Proc. of Adaptive Distributive Parallel Computing Symposium, Ohio, 1996.

[3] Guo Yiping and B.C.Forster, Unsupervised Classification of High Spectral Resolution Images Using the Kohonen Self-Organization Neural Network, Chinese Journal of Infrared and Millimeter Waves, vol.13, no.6, pp.457-465, 1994.

ニューラルネットワークのグラフ分割問題への応用

1 はじめに

グラフ分割問題は、グラフ上の点集合をいくつかの部分集合に分け、各部分集合に含まれる点の重みの総和をできるだけ等しく保ち、かつカットされる枝の重みを小さくする問題である。その応用分野はVLSI回路 [1] の設計や分散計算機環境におけるタスクの割り当てなど多岐にわたる。グラフ分割問題はNP完全問題 [2] として知られており、準最適解を求める多くの発見的手法が提案されている。KL(Kirnighan-Lin) アルゴリズム [3]、SA(Simulated Annealing)[4] 法、遺伝的アルゴリズム (Genetic Algorithm)[5] などが代表的な手法である。

本論文ではマキシマムニューロンモデル [6] による点 (node) と枝 (edge) に重みを持った無向グラフの分割手法を提案する。ここではAhmad等によって提案されたPSGA(Problem Space Genetic Algorithm) による手法 [5] との比較を行った。グラフ分割問題ではSA法によって良い解を得られることが知られているが、PSGAはそれよりも良い解を出すことが報告されており、また点と枝に重みのあるグラフを扱っていることから比較の対象として取り上げた。

2 グラフ分割問題の定義

点と枝に重みを持った無向グラフを $G=(V,E)$ とする。ここで V は点の集合であり $|V|=n$ (n は点の数)、E は枝の集合であり $E \subseteq V \times V$ である。点 v から点 u までの枝は $e_{v,u} \in E \ \forall u,v | u,v \in V$ で表される。点の重み $w(v)$ は $w(v) \in Z^+ \ \forall v | v \in V$ で定義される。ここで Z^+ は正の整数である。枝の重み $l(e_{v,u})$ は $l(e_{v,u}) \in Z^+$ で定義される。

グラフ分割問題は、点集合 V を m 個の部分集合 $V_1, V_2, ..., V_m$ に分ける問題である。ここで、$V_1 \cup V_2 \cup ... \cup V_m = V$ かつ $V_i \cap V_j = \phi \ (i \neq j$ であるとする。部分集合に含まれる点の重みの総和を $S(i) = \sum_{v \in V_i} w(v)$ とし、2つの部分集合間でカットされた枝の重みの総和を $C_{i,j} = \sum_{u \in V_i, v \in V_j} l(e_{u,v}) \ (i \neq j)$ とする。各部分集合における $S(i)$ の差の総計を $W_1 = \sum_{1 \leq i < j \leq m} |S(i) - S(j)|$ と定義し、カットされた枝の重みの総計を $W_2 = \sum_{1 \leq i < j \leq m} C_{i,j}$ と定義する。

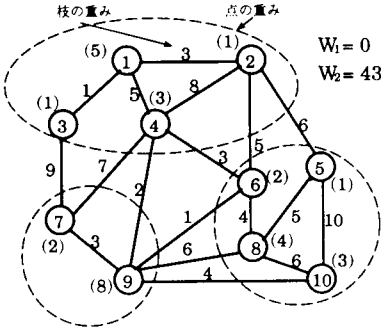

図 1: 分割例（3分割）

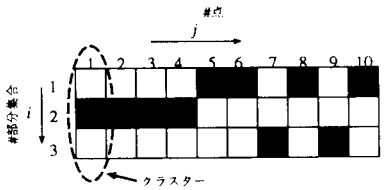

図 2: ニューラル表現

グラフ分割の目的は、W_1 を最小化しつつ、W_2 を最小化する分割を求めることにある。ここでは、PSGAとの比較を行うために、解を $Cost = \lambda_1 W_1 + \lambda_2 W_2$ で評価する。今回は $\lambda_1 = \lambda_2 = 1$ とした。図:1に10点グラフの分割例を示す。

3 ニューラル表現

グラフ分割問題のためのニューラル表現を図:2に示す。ここでは $m \times n$ 個のニューロンを用いる。ここで m は部分集合の数、n は点の数である。黒い部分は i,j 番目のニューロンが発火していることを示す。すなわち、ニューロンの出力が1であることを示している。これは、j 番目の点が i 番目の部分集合に含まれることを表している。白い部分は発火していないことを示す。すなわち、ニューロンの出力が0である。

ニューロンの入出力関数としてマキシマムニューロンモデルを採用した。マキシマムニューロンは、1つのクラスターの中に複数のニューロンが含まれているモデルで、そのなかのただ1つのニューロンが発火する。これを関数で表すと以下の通り。

$$V_{i,j} = \begin{cases} 1, & U_{i,j} = max(U_{x,j})(x = 1, ..., m) \\ 0, & otherwise \end{cases} \quad (1)$$

Uはニューロンへの入力、Vは出力を表す。図:2に示すように、この問題においてはj列に含まれるニューロンを1つのクラスターとする。

4 動作式

i,j番目のニューロンと他のニューロンとの接続は動作式によって表現される。動作式は問題の制約条件から決定される。グラフ分割問題のための動作式は以下の通り。

$$\begin{aligned}\frac{dU_{i,j}}{dt} &= -A\left(\sum_{k=1}^{n} w_k V_{i,k} - \frac{\sum_{k=1}^{n} w_k}{m}\right) \\ &- B\left(\sum_{p=1,p\neq i}^{m}\sum_{q=1,q\neq j}^{n} d_{j,q}V_{p,q}\right) \\ &+ C\left(\sum_{k=1}^{n} d_{j,k}V_{i,k}\right) + Dh\left(\sum_{k=1}^{n} V_{i,k}\right) \quad (2)\end{aligned}$$

A、B、Cはそれぞれ定数。

第1項は各部分集合の重みの総和のバランスをとるための項である。W_xは各点の重みである。理想的な状態は各部分集合の重みの総和$S(i)$が等しくなる、すなわちすべての点の重みの総和の平均$\frac{\sum_{k=1}^{n} w_k}{m}$をとることである。$S(i)$が平均より大きいときはニューロンの発火を抑制し、小さいときはニューロンの発火を促す働きをしている。

第2項は、i番目の部分集合においてj番目の点と他の部分集合における点との間でカットされた枝の重みに関する項である。$d_{x,y}$は点xと点yとの間の重みの（対称）行列である。この項ではカットされた枝の重みの合計に応じて発火を抑制する。

第3項はより良い解を得るために付け足された項で、カットされていない枝の重みの合計値である。この値に応じてニューロンの発火を促す。

第4項は、i番目の部分集合にニューロンが1つも発火していないときに強制的にニューロンを発火させるための項で、ヒルクライミング (Hill-Climbing) 項 [6] と呼ばれているものである。ここで関数hは$h(X) = 1\ if\ X = 0,\ 0\ otherwise$である。また$D = 4\ if\ t\ modulo\ 10 < 4,\ 1\ otherwise$とする。

動作式では、それぞれのコストは部分的に計算されている。総コスト$Cost$を計算するためにW_1を$W_1 = \sum_{1\leq x<y\leq m} |\sum_{k=1}^{n} w_k V_{x,k} - \sum_{k=1}^{n} w_k V_{y,k}|$で$W_2$を$W_2 = \sum_{x=1}^{n}\sum_{y=x+1}^{n}\sum_{z=1}^{m}\sum_{k=1,k\neq z}^{m} d_{x,y}V_{z,x}V_{k,y}$でそれぞれ計算する。

5 アルゴリズム

以下にアルゴリズムを示す。t_limitは終了条件として、繰返し数の最大値を示し、$target_cost$は目標とする総コストである。

1. 繰返し数tを0にセットし、A, B, C, t_limit, $target_cost$の値を決める。
2. ニューロンの入力$U_{i,j}(t)$ $(i = 1,...,m,\ j = 1,...,n)$の値をランダムに決める。
3. $j = 1,...,n$に対して、
   (a) 動作式を用いて$V_{i,j}(t)$ $(i = 1,...,m)$を評価する。
4. 総コスト$(Cost)$を計算する。もし$target_cost < Cost$ならば終了。
5. tに加算する。
6. $j = 1,...,n$に対して、
   (a) 動作式を計算して$i = 1,...,m$に対してそれぞれ$\Delta U_{i,j}(t)$を計算する。
   (b) $i = 1,...,m$に対して1次のオイラー法を用いて$U_{i,j}(t+1)$の値を更新する：
   $U_{i,j}(t+1) = U_{i,j}(t) + \Delta U_{i,j}(t)$
   (c) $i = 1,...,m$に対して$V_{i,j}(t+1)$を式(1)を用いて評価する.
7. $Cost$を計算する。もし$target_cost > Cost$かt=t_limitなら終了。さもなくばtに1を加算してstep 6へ。

6 シミュレーションとその結果

提案されたアルゴリズムはC言語で実装され、DEC Alpha Station $200^{4/166}$上でシミュレーションを行った。

シミュレーションには点と枝に重みを持つランダムグラフ [4] を用いた。ランダムグラフの枝は、確率$p(0 < p < 1)$で生成された。ノードあたりの枝の次数 (Degree) の期待値は$p(n-1)$である。シミュレーションでは点の数50,100,150のグラフでそれぞれ次数を変えたものを用いた。点の数50のグラフでは重みは1から10の整数値をそれ以外では1から20の整数値をそれぞれランダムに与えた。

表1に結果を示す。シミュレーションでは動作式のAの値を2.0,3.0,4.0にそれぞれ変えて行った。B, Cには$B = C = \frac{1}{\sum_{k=1}^{n} d_{j,k}}$を用いた。PSGAに関しては文献 [5] と同じ設定にした。表の結果は各シミュレーションで総コストが最小のものと計算時間 (CPU time) である。

表 1: シミュレーション結果

				Neural Network				PSGA			
Node	Degree	Edge	m	W_1	W_2	*Cost*	time(s)	W_1	W_2	*Cost*	time(s)
50	4	98	4	3	163	166	0.78	3	263	266	4.05
			6	15	216	231	0.98	25	326	351	3.58
			8	7	270	277	0.77	19	363	382	3.45
50	8	198	6	13	633	646	0.05	5	790	795	2.15
			8	53	663	716	0.03	19	827	846	2.20
			10	49	768	817	0.05	33	878	911	4.75
50	12	312	4	13	899	912	1.12	19	1044	1063	3.10
			8	37	1210	1247	0.07	7	1359	1366	5.13
			12	101	1345	1446	0.07	11	1490	1501	5.07
100	8	398	6	58	1998	2056	0.83	26	3139	3165	2.77
			8	104	2264	2368	0.12	26	3139	3165	8.55
			10	104	2494	2598	0.13	70	3228	3298	4.13
			15	392	2616	3008	0.15	146	3477	3623	8.20
100	12	519	6	76	3548	3624	0.13	30	4470	4500	3.85
			8	86	3984	4070	0.12	118	4767	4885	8.63
			10	82	4191	4273	0.13	82	5024	5106	11.55
			15	390	4545	4935	0.20	146	5317	5463	12.37
100	16	819	6	78	5420	5498	0.13	50	6477	6527	5.33
			8	84	5978	6062	0.15	54	6915	6969	12.25
			10	270	6117	6387	0.17	58	7219	7277	5.00
			15	334	6844	7178	0.22	100	7620	7720	5.17
150	8	594	6	82	2789	2871	3.78	42	4815	4875	15.16
			8	88	3263	3351	5.51	80	4841	4921	15.80
			10	106	3601	3707	0.20	98	4955	5053	13.13
			15	320	4020	4340	0.30	108	5270	5378	15.58
150	12	894	6	90	5514	5604	0.22	22	7146	7168	16.23
			8	106	6050	6156	0.27	82	7594	7676	16.28
			10	134	6495	6629	0.22	122	7906	8028	12.65
			15	388	6943	7331	0.28	134	8327	8461	18.81
150	16	1193	6	70	7716	7786	0.33	88	9483	9571	12.63
			8	114	8381	8495	0.25	78	10096	10174	17.43
			10	308	8747	9055	0.25	48	10541	10589	13.38
			15	506	9529	10035	0.35	246	10977	11223	7.88

7 結論

表 1をみると、ニューラルネットワークの方が短時間に小さなカットの解を出していることがわかる。それに比べ PSGA は特に分割数が大きくなったときにニューラルネットワークよりも W_1の値が小さい解を出すことがわかる。W_1の小さな値を安定して出すように改良することが今後の課題である。

参考文献

[1] L. Tao and Y.C. Zhao, "*Effective heuristic algorithms for VLSI circuit partition*," IEE Proceedings-G, Vol.140, No.2, April 1993, 127-134.

[2] M.R. Garey and D.S. Johnson, "*Computers and Intractability: A Guide to the Theory of NP-Completeness*," W.H. Freeman, New York, 1979.

[3] B.W. Kernighan and S. Lin, "*An Efficient Heuristic Procedure for Partitioning Graphs*," The Bell Sys. Tech. J., February. 1970, 291-307.

[4] D.S. Johnson, C.R. Aragon, L.A. McGeoch, and C. Schevon, "*Optimization by Simulated Annealing: An Experimental Evaluation; Part I, Graph Partitioning*," Operations Research, Vol.37, No.6, November-December, 1989, 865- 892.

[5] I. Ahamd and M.K. Dhodhi, "*On the m-Way Graph Partitioning Problem*," The Computer Journal, Vol.38, No.3, 1995, 237-244.

[6] Y. Takefuji, "*Neural Network Parallel Computing*, " Kluwer Academic Publishers, 1992.

[7] Y. Takenaka, N. Funabiki and S. Nishikawa, "*Maximum Neural Network Algorithm for N-Queen Problems*," Transactions of Information Processing Society of Japan, Vol. 37, No. 10, October 1996, 1781-1788.

[8] T. Saito and Y. Takefuji, "Neural Computing for the m-Way Graph Partitioning Problem", IEICE Transactions on Information. and Systems., Vol.E80-D, No.9, September 1997, pp.942-947.

研究成果リスト (1995 年～1997 年)

[1] 「ニューラルネットワークによる並列処理のおもしろさ」, 武藤佳恭, 第 3 回 農林水産省 農業環境技術研究所 計測と情報解析研究会, 1995.

[2] K. Tsuchiya and Y. Takefuji, "A neural network algorithm for the no-three-in-line problem", Neurocomputing, vol.8, no.1, pp. 43-9, 1995.

[3] K. Fujisawa and Y. Takefuji, "A balloon net discovering improved solutions in one of unsolved problems in geometry: a problem of spreading points in a unit square", 1995 IEEE International Conference on Neural Networks Proceedings (Cat. No.95CH35828), no. 6 vol. 1+3219, pp.2208-10.

[4] T. Saito and Y. Takefuji, "Neural computing approach to Japanese electoral system", 1995 IEEE International Conference on Neural Networks Proceedings (Cat. No.95CH35828), no. 6 vol. 1+3219, pp.2202-7.

[5] K. Suzuki, H. Amano, and Y. Takefuji "Neural network parallel computing for multi-layer channel routing problems", Neurocomputing, vol.8, no.2, pp. 141-56, 1995.

[6] S. Y. Foo, Y. Takefuji, and H. Szu, "Scaling properties of neural networks for job-shop", Neurocomputing, vol.8, no.1, pp. 79-91, 1995.

[7] K. Tsuchiya, S. Bharitkar and Y. Takefuji, "A neural network approach to facility layout problems", European Journal of Operational Research, vol.89, no.3, pp. 556-63 , 1996.

[8] T. Ogawa, S. Oka, Y. Takefuji, and T. Yao, "A Functional-link Net Algorithm for Pattern Recognition Problems of Super Secondary Structures in Protein Sequence", Proc. of Adaptive Distributed Parallel Computing, August 8-9, 1996.

[9] S. Oka, T. Ogawa, T. Oda and Y. Takefuji, "A New Self-Organization Classification Algorithm for Remote-Sensing Data", Proc. of Adaptive Distributed Parallel Computing, August 8-9, 1996.

[10] K. Tsuchiya and Y. Takefuji, "A neural network approach to PLA folding problems, IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems", vol.15, no.10, pp. 1299-305, 1996.

[11] N. Oshima, T. Ogawa, and Y. Takefuji, "Airport allocation problems in Mongolia using neural networks", Proc. of the Eighth Australian Conference on Neural Networks (ACNN'97), pp. viii+201, 197-201, 1997.

[12] M. Aoba and Y. Takefuji, "Improved neural algorithms for knight's tour problems", Proc. of Proc. of the Eighth Australian Conference on Neural Networks (ACNN'97), pp. viii+201, 163-6, 1997.

[13] Y. Takefuji and S. Oka, "Innovations in Materials Design on the Information Highway: Electronic Prototyping - Classification, Discretization, and Functional Mapping", Proc. of Pacific forum on Intelligent Processing and Manufacturing of Materials, Australia, 1997.

[14] Yoshiyasu Takefuji, "Neural Computing for Optimization and Combinatorics", World Scientific Publisher, 1996.

[15] 武藤佳恭, 岡宗一 ,「難問への挑戦」, 電子情報通信学会誌, 教養のページ, 10 月号, 1996.

[16] 斉藤孝之, 武藤佳恭,「ニューラルコンピューティングによる小選挙区区割り手法」, 情報処理学会論文誌, 37 巻 4 号, 588-596 ページ, 1996.

[17] 武藤佳恭, 「ニューラルコンピューティング」, コロナ社, 1996.

[18] 武藤佳恭,「ニューラルネットワーク」, 産業図書, 1996.

[19] 武藤佳恭他, "bit ニューラルコンピューティングの遊び方", 共立出版, 1995～1997.