X Congresso Brasileiro de Inteligência Computacional (CBIC'2011), 8 a 11 de Novembro de 2011, Fortaleza, Ceará

ALGORITMO HÍBRIDO PARA ESTIMAÇÃO ML DE MODELOS HMM BASEADO EM TREINAMENTO DE VITERBI E OTIMIZAÇÃO PSO

Elaine C. Marques, Nilson Maciel e Ernesto L. Pinto
Instituto Militar de Engenharia, Seção de Ensino de Engenharia Elétrica, Laboratório de Comunicações Digitais,
{elainecmarques,nilsonmpj,ernesto}@ime.eb.br

Resumo – Propõe-se aqui um algoritmo híbrido baseado no Treinamento de Viterbi e no algoritmo PSO (Otimização por Enxame de Partículas) para a estimação de máxima verossimilhança dos parâmetros dos modelos HMM. Uma comparação de desempenho com os algoritmos Baum-Welch (BW) e PSO é apresentada, focando principalmente a habilidade dos algoritmos em se aproximar da solução de máximo global. Os resultados obtidos sugerem que o algoritmo proposto tem desempenho superior aos algoritmos BW e PSO.

Palavras-chave – HMM, estimação ML, Treinamento de Viterbi, PSO, erros em surtos.

Abstract – A hybrid algorithm based on Viterbi Training and Particle Swarm Optimization (PSO) is here proposed to perform maximum-likelihood estimation of HMM models. A performance comparison with the Baum-Welch (BW) algorithm and with the PSO algorithm is presented, which is focused on the ability of the algorithms to search for the global solution. The numerical results obtained suggest that the proposed algorithm outperforms both the BW and the PSO algorithms.

Keywords – HMM, ML estimation, Viterbi Training, PSO, burst errors.

1. INTRODUÇÃO

Os Modelos de Markov Escondidos (HMM, de *Hidden Markov Model*) têm sido aplicados em diversas áreas de pesquisa, tais como reconhecimento de locutor [1], processamento de imagem e visão computacional [2], reconhecimento de texto [3], canais de comunicações com erros em surtos [4], entre outros [5–8].

Para ajuste dos parâmetros destes modelos é de grande interesse o emprego do método de máxima verossimilhança (ML, de "maximum likelihood"), sendo o algoritmo Baum-Welch (BW) frequentemente empregado para este fim [9]. Este algoritmo é reconhecidamente capaz de obter um valor maior da função de verossimilhança a cada iteração, mas não tem convergência assegurada para o máximo de verossimilhança global. Além disso, o algoritmo BW também tem complexidade computacional elevada [10].

Assim sendo, a investigação de algoritmos alternativos para estimação ML de modelos HMM, com maior facilidade de alcançar a solução ML global e/ou com menor custo computacional, tem despertado o interesse da comunidade científica internacional ao longo dos últimos anos [11–13].

O Treinamento baseado no algoritmo de Viterbi (VT), por exemplo, tem sido considerado uma alternativa ao algoritmo BW para a estimação de modelos HMM em diversas aplicações, tais como reconhecimento de voz, análise de imagens e bioinformática [14], e sistemas de comunicações com erros em surtos [15].

O VT não produz necessariamente aumento no valor da verossimilhança a cada iteração, mas tem complexidade computacional muito menor do que o BW [10]. Uma comparação do desempenho destes dois algoritmos na estimação ML de modelos HMM foi feita em [15], considerando-se canais com erros em surtos, a qual indicou que o VT pode produzir boas estimativas em tempos de processamento significativamente inferiores aos do BW.

No que diz respeito especificamente à maior facilidade de alcançar a solução ML global na estimação de modelos HMM, alguns trabalhos recentes têm focado na aplicação de algoritmos de otimização de funções não lineares baseados em metaheurísticas, com destaque para o emprego do algoritmo de Otimização por Enxames de Partículas (PSO, de *Particle Swarm Optimization*) [12, 13, 16–18].

Em [19] foi feita uma comparação entre o desempenho dos algoritmos BW e PSO na estimação dos parâmetros dos modelos HMM aplicados em canais com erros em surtos e se constatou que de fato o algoritmo PSO pode produzir valores de verossimilhança significativamente melhores do que o algoritmo BW, para um mesmo tempo de processamento.

No presente trabalho procura-se avançar na busca de ferramentas computacionais para estimação ML dos parâmetros de modelos HMM, propondo-se um algoritmo híbrido baseado nos algoritmos PSO e VT, o qual será doravante denominado VTPSO.

É também apresentada neste artigo uma comparação de desempenho entre o algoritmo proposto e os algoritmos PSO e BW, na estimação de modelos HMM para canais com erros em surtos. Os resultados desta comparação sugerem que o algoritmo VTPSO apresenta desempenho superior, conseguindo aparentemente reunir características vantajosas do VT e do PSO.

O restante deste trabalho está organizado como se segue: a seção 2 apresenta alguns conceitos básicos de modelagem HMM; a seção 3 descreve os algoritmos BW, VT e PSO; o algoritmo proposto é apresentado na seção 4; o ajuste de modelos HMM para

erros em surtos é discutido na seção 5; os resultados das avaliações de desempenhos são apresentados na seção 6 e a seção 7 conclui o trabalho.

2. MODELOS HMM

Um modelo HMM é caracterizado pelo número de estados e pelas distribuições de probabilidades condicionais de transição de estados e das observações, dados os estados. Admitindo-se que estas observações assumem valores em um conjunto discreto e finito, tem-se os seguintes parâmetros:

- N - Número de estados da cadeia de Markov;
- m - Número de símbolos do alfabeto de observações;
- A - Matriz das probabilidades de transições de estados, definida na Equação (1), onde q_t representa o estado no índice de tempo t. As probabilidades de transição $a_{i,j}$ devem atender às condições apresentadas na Equação (2);

$$A = \{a_{i,j}\};\ \ a_{i,j} = p(q_{t+1} = j | q_t = i);\ \ 1 \leq i,\ \ j \leq N \tag{1}$$

$$0 \leq a_{i,j} \leq 1; \sum_{j=1}^{N} a_{i,j} = 1; 1 \leq i, j \leq N; \tag{2}$$

- B - Matriz das distribuições condicionais de probabilidades das observações dados os estados, expressa na Equação (3), onde v_k representa o k-ésimo símbolo de observação no alfabeto e o_t a observação no índice de tempo t. As probabilidades b_j devem satisfazer às condições apresentadas na Equação (4).

$$B = \{b_j(k)\};\ \text{ onde: } b_j(k) = p(o_t = v_k | q_t = j)\ \ 0 \leq j \leq N;\ \ 1 \leq k \leq m \tag{3}$$

$$0 \leq b_j(k) \leq 1;\ \ \sum_{k=1}^{m} b_j(k) = 1;\ \ 0 \leq j \leq N;\ \ 1 \leq k \leq m \tag{4}$$

- Π - Vetor das probabilidades de inicialização em cada estado, dada pela Equação (5). As probabilidades Π_i devem obedecer às condições apresentadas na Equação (6).

$$\Pi = \{\Pi_i\}; \qquad \text{onde:} \qquad \Pi_i = p(q_1 = i) \tag{5}$$

$$0 \leq \Pi_i \leq 1; \quad \sum_{i=1}^{N} \Pi_i = 1; \quad 1 \leq i \leq N \tag{6}$$

Denota-se aqui de maneira compactada os parâmetros de um modelo HMM deste tipo por $\lambda = (A, B, \Pi)$.

3. ALGORITMOS DE ESTIMAÇÃO

Nesta seção são apresentados três algoritmos de estimação previamente propostos que são utilizados neste trabalho.

3.1 Algoritmo Baum-Welch (BW)

O algoritmo Baum-Welch [20] tem sido a ferramenta mais utilizada para a estimação de parâmetros de modelos HMM. É um caso particular do bem conhecido algoritmo EM para a estimação ML e pode ser descrito pelos seguintes passos:

Passo 0: Inicializar a estimativa de parâmetros $\hat{\lambda} = \{\hat{A}, \hat{B}, \hat{\Pi}\}$.

Passo 1: Calcular as variáveis progressivas e regressivas, utilizando $\hat{\lambda}$.

$$\alpha_t(i) = P[O_1, O_2, \ldots, O_t, s_t = i | \hat{\lambda}] \quad \beta_t(i) = P[O_{t+1}, O_{t+2}, \ldots, O_T | s_t = i, \hat{\lambda}] \tag{7}$$

Passo 2: Atualizar $\hat{\lambda}$, usando α_t e β_t obtidos no Passo 1. As expressões para esta atualização podem ser obtidas em [20].

Passo 3: Verificar a condição de parada. Se esta não for atendida, retornar para o Passo 1 com a atualização de $\hat{\lambda}$.

3.2 Treinamento baseado no Algoritmo de Viterbi (VT)

Este método possibilita estimar recursivamente as matrizes A e B do modelo[1] e consiste basicamente em obter a sequência de estados com máxima probabilidade a posteriori (MAP) associada às estimativas atuais de A e B, e usá-la para atualizar estas estimativas. O VT pode ser descrito pelos seguintes passos:

Passo 0: Inicializar as estimativas $\hat{A}$ e $\hat{B}$.

Passo 1: Estimar a sequência de estados de máxima probabilidade a posteriori (MAP), usando o algoritmo de Viterbi.

Passo 2: Atualizar as estimativas $\hat{A}$ e $\hat{B}$ pelo cálculo de frequências relativas de transição de estados e de frequências relativas de observações a partir de cada estado, usando a sequência MAP.

Passo 3: Verificar a condição de parada e, se necessário, retornar para o Passo 1 com os novos valores de $\hat{A}$ e $\hat{B}$.

[1] A estimativa do vetor Π é calculada a partir da estimativa de A, admitindo-se que o modelo é iniciado em regime estacionário.

3.3 Otimização por Enxame de Partículas (PSO)

O PSO é um algoritmo de otimização não linear baseado numa analogia sócio-cognitiva simples, segundo a qual cada indivíduo de uma população é capaz de quantificar a qualidade de sua experiência na busca de um determinado objetivo, e também tem conhecimento dos resultados prévios obtidos por seus vizinhos. Admite-se ainda que um determinado indivíduo decide sobre como prosseguir nesta busca com base nos melhores resultados obtidos previamente por ele mesmo e pelo conjunto de seus vizinhos [16].

Com base nesta analogia, a cada iteração o algoritmo gera e atualiza uma quantidade pré-fixada de amostras dos parâmetros da função a ser otimizada. Cada uma destas amostras é usualmente denominada "partícula" e o seu conjunto é chamado de "enxame". Uma dada partícula i é representada por um vetor de posição $\overrightarrow{x_i}(t)$ e por um vetor de velocidade instantânea $\overrightarrow{v_i}(t)$. Além disso, o algoritmo retém a posição $\overrightarrow{p_i}$ em que cada partícula obteve o melhor valor da função custo, assim como a posição $\overrightarrow{p_g}$ correspondente ao melhor resultado obtido por todo o enxame.

A velocidade e a posição de uma dada partícula são atualizadas segundo as equações a seguir:

$$\overrightarrow{v_i}(t+1) = W\overrightarrow{v_i}(t) + c_1U_1(\overrightarrow{p_i}(t) - \overrightarrow{x_i}(t)) + c_2U_2(\overrightarrow{p_g}(t) - \overrightarrow{x_i}(t)) \quad (8)$$

$$\overrightarrow{x_i}(t+1) = \overrightarrow{x_i}(t) + \alpha\overrightarrow{v_i}(t+1) \quad (9)$$

onde o parâmetro W, chamado de peso de inércia, regula a influência do último valor de velocidade sobre o atual, U_1 e U_2 são valores uniformemente distribuídos no intervalo $[0\ \ 1]$, as constantes c_1 e c_2 ponderam a influência da experiência própria passada e social, respectivamente, e α pondera o efeito da velocidade na atualização da posição.

As Equações (8) e (9), assim como os valores de $\overrightarrow{p_i}$ e $\overrightarrow{p_g}$, são calculados a cada iteração até que seja satisfeito o critério de parada adotado (por exemplo, um número máximo de iterações permitidas).

4. ALGORITMO PROPOSTO

O algoritmo VTPSO aqui proposto procura reunir características vantajosas dos algoritmos PSO (maior flexibilidade para buscar a solução de máximo global) e VT (simplicidade), a fim de ser capaz de gerar soluções com alto valor de verossimilhança em pouco tempo de processamento. Esta proposta pode ser vista como uma evolução do algoritmo PSO na qual a atualização de cada partícula, além de sofrer as influências da experiência própria passada e da experiência social, também é influenciada por uma informação produzida pelo algoritmo VT.

A atualização das partículas no algoritmo VTPSO é realizada mediante a aplicação das Equações (10) e (11), onde $\overrightarrow{pv_i}$ representa a referência de posição obtida após um número pré-fixado (X) de iterações do VT e a constante c_3 regula o peso desta referência no processo de atualização. Os demais parâmetros destas equações têm o mesmo significado que teriam no algoritmo PSO convencional.

$$\overrightarrow{v_i}(t+1) = W\overrightarrow{v_i}(t) + c_1U_1(\overrightarrow{p_i}(t) - \overrightarrow{x_i}(t)) + c_2U_2(\overrightarrow{p_g}(t) - \overrightarrow{x_i}(t)) + c_3U_3(\overrightarrow{pv_i}(t) - \overrightarrow{x_i}(t)) \quad (10)$$

$$\overrightarrow{x_i}(t+1) = \overrightarrow{x_i}(t) + \alpha\overrightarrow{v_i}(t+1) \quad (11)$$

A fim de garantir que o movimento da partícula se dê na direção indicada na Equação (10) e ao mesmo tempo atender às restrições estocásticas dos modelos HMM, o valor de α é calculado segundo o procedimento apresentado em [21].

Denotando por $f(a)$ o valor da função custo na posição a, o VTPSO pode ser descrito pelos seguintes passos:

Passo 0: Inicializar as partículas, escolhendo aleatoriamente ($\overrightarrow{x_i}(0)$) e ($\overrightarrow{v_i}(0)$). Fazer $f(\overrightarrow{p_i}(t)) = -\infty$ e $f(\overrightarrow{p_g}(t)) = -\infty$.

Passo 1: Para cada partícula, verificar se $f(\overrightarrow{x_i}(t))$ é maior que $f(\overrightarrow{p_i}(t))$. Caso seja, atualizar $\overrightarrow{p_i}(t)$.

Passo 2: Verificar se $f(\overrightarrow{p_i}(t))$ é maior que $f(\overrightarrow{p_g}(t))$, para cada partícula. Se for, atualizar $\overrightarrow{p_g}(t)$.

Passo 3: Atualizar os valores de $\overrightarrow{x_i}(t+1)$ e $\overrightarrow{v_i}(t+1)$ de cada partícula, usando as Equações (10) e (11).

Passo 4: Verificar a condição de parada e, se necessário, retornar para o Passo 1.

5. MODELAGEM DE ERROS EM SURTOS

A ocorrência de erros em surtos é um problema de grande interesse na área de telecomunicações, e pode estar relacionada com o meio de propagação, com o tipo de ruído presente no sistema (ruído impulsivo), com interferências ou com o uso de técnicas de recepção com memória [4, 22, 23].

A investigação do impacto destes no desempenho de sistemas e redes de comunicações é um campo de pesquisa bastante atual, no qual é de grande importância o estabelecimento de modelos matemáticos adequados [22, 23].

Neste contexto, a aplicação de modelos HMM é potencialmente vantajosa, tanto para análise matemática de desempenho quanto para investigações baseadas em simulações. O ajuste de parâmetros de modelos desta classe a amostras de erros empíricas sobressai então como um problema relevante, o qual está ilustrado na Figura 1.

O ponto de partida é um canal de comunicação com uma fonte de erros que produz uma sequência de erros a ser empregada no ajuste. Esta sequência é composta por bits "0", indicando decisões corretas e bits "1" que indicam a ocorrência de erros.

A fim de iniciar o processo de ajuste, um modelo HMM candidato é inicialmente estabelecido. Em seguida, os parâmetros deste modelo são estimados, usando um algoritmo específico. Com esta atribuição de parâmetros, o modelo candidato é utilizado para gerar uma "sequência de erros de teste", a partir da qual são obtidas estimativas de diversas estatísticas comumente utilizados

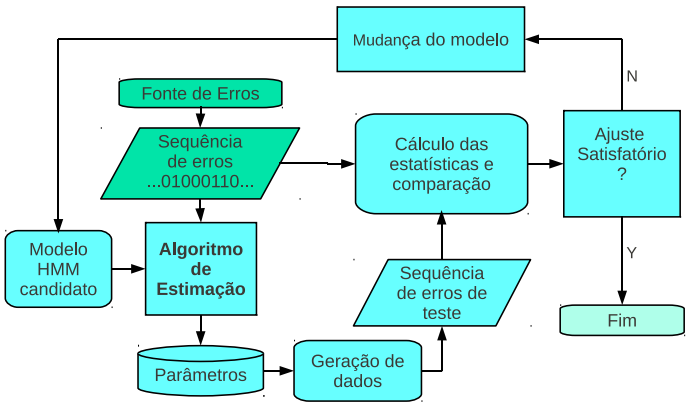


Figura 1: Ajuste do modelo HMM a amostra de erros em surtos.

para descrever processos de erros em surtos (distribuição de comprimentos de surtos, distribuição de comprimentos de intervalos entre surtos, etc.) [22]. Estas estimativas são então comparadas com estimativas similares extraídas da sequência de erros original. Se ocorrer um ajuste satisfatório, o modelo candidato é aceito. Caso contrário, é feita uma mudança na sua estrutura, como por exemplo o número de estados, e o novo modelo HMM candidato assim obtido é avaliado de forma semelhante. Este ciclo deve ser repetido até que uma aproximação satisfatória das estatísticas extraídas da sequência de erros original seja alcançada.

Vale ressaltar que podem ser necessário testar vários modelos candidatos até se encontrar um ajuste satisfatório. Assim sendo, o algoritmo empregado para estimar os parâmetros destes modelos desempenha um papel crucial no processo de ajuste, tendo impacto significativo na sua eficácia e na sua complexidade computacional.

6. RESULTADOS

A seguir são apresentados resultados de uma avaliação do desempenho do algoritmo proposto na aplicação à modelagem de erros em surtos, que inclui uma breve comparação de desempenho com os algoritmos PSO e BW.

Nesta avaliação tomou-se como principal parâmetro de desempenho a distribuição de valores da função de verossimilhança obtidos após a execução dos algoritmos por um número elevado de vezes, sobre uma mesma amostra (sequência de erros) e com diversas inicializações geradas aleatoriamente. As simulações foram feitas em computadores Core 2 Quad CPU 2.66GHz e os algoritmos foram implementados usando Sage [24].

Para apresentar os resultados são utilizadas curvas em que se tem como abscissa o percentual das inicializações (denotado por INI) que levou a valores de logaritmo da função de verossimilhança superiores à ordenada (denotado por LL). Curvas deste tipo serão doravante denominadas distribuições de verossimilhanças ou distribuições de valores de verossimilhanças.

Para levantamento das distribuições de verossimilhança considerou-se o ajuste dos parâmetros de um modelo HMM de 3 estados a uma sequência de erros de tamanho 10^6 produzida por um sistema de transmissão QPSK num canal Rayleigh com espectro Doppler de Jakes e espalhamento Doppler normalizado $f_D T = 10^{-4}$. Admitiu-se um receptor operando com sincronização perfeita, e razão $E_b/N_0 = 20dB$ na sua entrada. Escolheu-se o número de 3 estados para agilizar a execução dos experimentos. Como visto em [15], o aumento do número de estado aumenta significativamente o tempo de processamento do algoritmo BW e seu desempenho com tempo de processamento pré-fixado pode piorar ainda mais em relação ao VT.

Em todas as simulações, os algoritmos PSO e VTPSO foram implementados com 30 partículas, e o parâmetro X do VTPSO foi fixado em 1 (ou seja, apenas 1 iteração do VT foi usada para gerar sua contribuição à atualização de uma dada partícula).

Os parâmetros $(W \;\; c_1 \;\; c_2)$ do VTPSO e do PSO convencional foram fixados em $(0,7 \;\; 2,0 \;\; 2,0)$, tomando por base trabalhos anteriores [13, 19].

6.1 Comparação de Desempenho

Inicialmente se procura fazer uma sondagem do potencial do algoritmo VTPSO para a obtenção de soluções de máxima verossimilhança, comparando-o com os algoritmos PSO e BW sob condições similares de tempo de processamento. Nesta sondagem inicial o valor do parâmetro c_3 do VTPSO foi fixado em $2,0$

Reconhecendo as limitações do emprego do tempo de processamento de um algoritmo como medida de sua complexidade computacional, optou-se por utilizá-la para uma abordagem inicial deste tópico, tomando uma série de cuidados para evitar distorções na comparação entre os algoritmos VTPSO, PSO e BW. Com este intuito foram excluídas da avaliação do tempo de processamento todas as operações de entrada e saída para arquivamento das simulações, bem como outras que poderiam influenciar erroneamente esta medida indireta da carga computacional dos algoritmos. Além disso, os códigos de implementação dos algoritmos foram produzidos por um mesmo programador. Por fim, os testes para aferição dos tempos de processamento foram realizados em condições rigorosamente idênticas, no que diz respeito a hardware, software e condições de execução.

Foram estabelecidas duas referências de esforço computacional, 620s e 1240s, correspondentes a 500 e 1000 iterações do BW, respectivamente. Verificou-se também em testes prévios as quantidades de iterações dos outros algoritmos avaliados que levariam a estes tempos de processamento. Os resultados estão apresentados na Tabela 1.

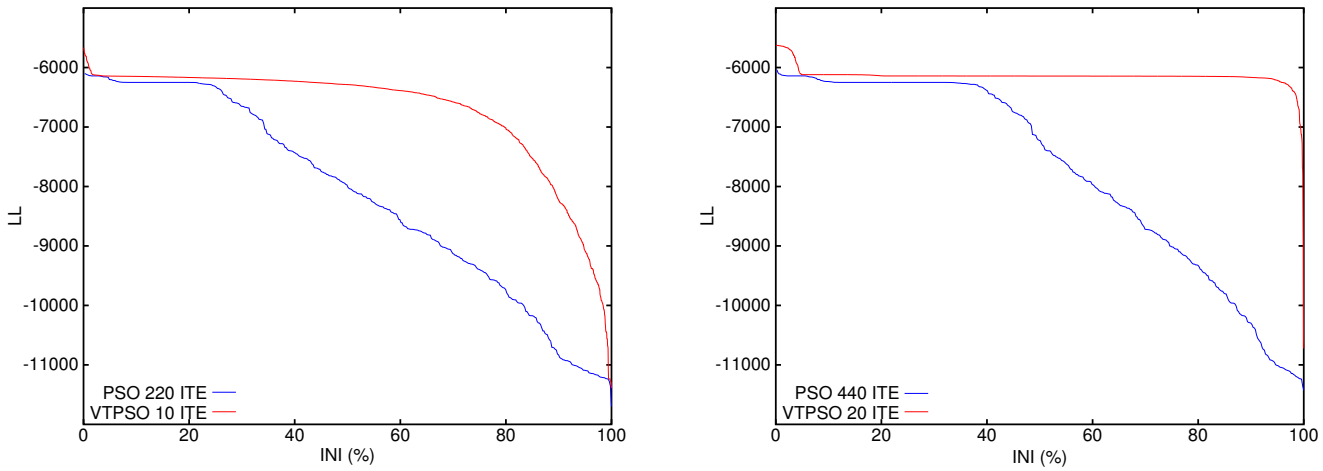
Tabela 1: Algoritmo × Iterações.

Algoritmo	Iterações em 620s	Iterações em 1240s
BW	500	1000
PSO	220	440
VTPSO	10	20

Foram em seguida realizadas 2300 execuções de cada um dos algoritmos, com inicializações aleatórias independentes, e se avaliou a distribuição dos valores de verossimilhança alcançados após as quantidades de iterações mostradas nesta tabela.

VTPSO* × *PSO

As Figuras 2(a) e 2(b) mostram as curvas de distribuição dos valores de verossimilhança obtidas com os algoritmos VTPSO e PSO com tempos de execução de 620 e 1240 segundos, respectivamente. Percebe-se que para os dois casos, o algoritmo VTPSO levou a percentuais muito mais elevados de valores de verossimilhança altos do que o PSO convencional, caracterizando-se assim uma indicação de maior eficácia na maximização da função de verossimilhança.



(a) (b)

Figura 2: Distribuições de verossimilhanças dos algoritmos VTPSO e PSO convencional para tempos de processamento de (a) 620 segundos e (b) 1240 segundos.

Tabela 2: LL × Inicializações.

Algoritmo	Iteração	≥-6000	≥-6500
PSO	220	0,0%	27,1%
PSO	440	0,0%	41,4%
VTPSO	10	1,0%	66,9%
VTPSO	20	4,3%	98,6%

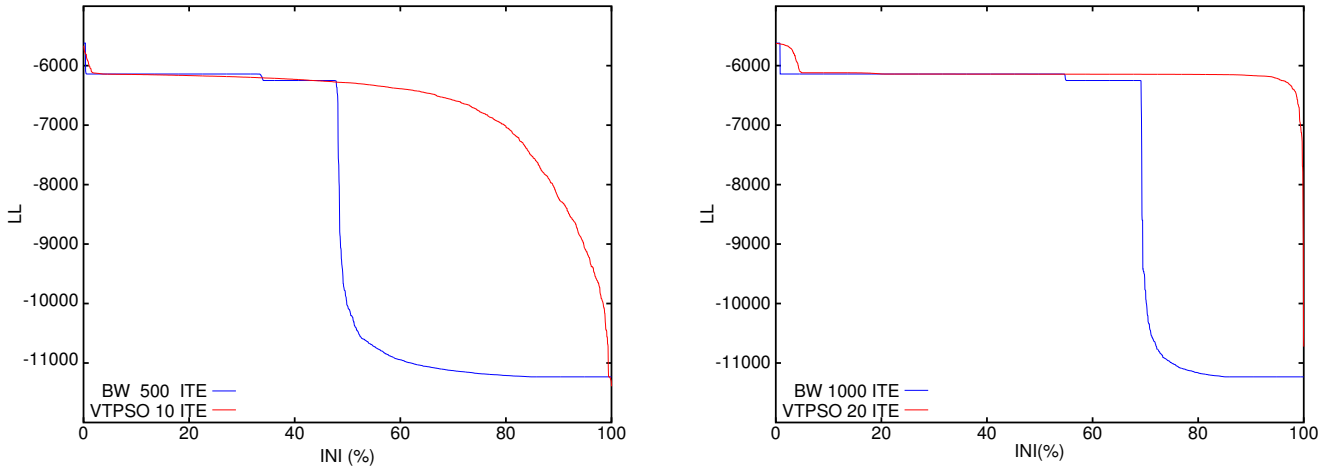
A mesma indicação se extrai da análise da Tabela 2, na qual se apresenta, para cada um dos casos, o percentual das 2300 execuções em que cada algoritmo produziu valores de LL acima de valores -6500 e de -6000.

VTPSO* × *BW

Nas Figuras 3(a) e 3(b) podem ser comparadas as curvas de distribuição de verossimilhança obtidas com os algoritmos VTPSO e BW para tempos de execução de 620 e 1240 segundos, respectivamente. Observa-se nestas figuras que o algoritmo VTPSO produziu concentração muito maior de valores elevados de verossimilhança do que o algoritmo BW, para ambos os valores do tempo de execução, maximizando a função de verossimilhança com maior eficácia.

A mesma característica pode ser verificada nos resultados apresentados na Tabela 3, onde se mostra, para cada um dos casos, o percentual das 2300 execuções em que cada algoritmo produziu valores de LL acima dos valores -6500 e -6000.

Os resultados apresentados indicam portanto que, além de poder alcançar valores de verossimilhança superiores ao do PSO convencional, o que pode ser atribuído à influência do treinamento baseado no algoritmo de Viterbi, o algoritmo proposto também é capaz de produzir valores de verossimilhança superiores aos obtidos com o algoritmo Baum-Welch. Parece razoável atribuir esta última vantagem à capacidade de não ficar "parado" em máximos locais, que o VTPSO tem em comum com o PSO convencional, em oposição ao algoritmo BW, o qual é reconhecidamente limitado neste aspecto.



(a) (b)

Figura 3: Distribuições de verossimilhanças dos algoritmos VTPSO e BW para tempos de processamento de (a) 620 segundos e (b) 1240 segundos.

Tabela 3: LL $\times$ Inicializações.

Algoritmo	Iteração	$\geq$-6000	$\geq$-6500
BW	500	0,3%	48,1%
BW	1000	0,8%	69,2%
VTPSO	10	1,0%	66,9%
VTPSO	20	4,3%	98,6%

6.2 Investigação adicional de desempenho do VTPSO

Nesta seção é feita uma primeira avaliação do efeito da variação do número de iterações e da constante c_3 (que pondera a influência do treinamento de Viterbi na atualização de velocidade) sobre o desempenho do algoritmo VTPSO.

Efeito do número de iterações

A Figura 4 apresenta as curvas de distribuição de verossimilhanças do VTPSO levantadas com 20, 50 e 100 iterações, mantendo-se o valor do parâmetro c_3 em 2,0. Como era de se esperar, com o aumento da quantidade de iterações, aumenta-se o percentual de realizações com maiores valores de verossimilhança (às custas de aumentos no tempo de processamento).

Percebe-se nesta figura que o logaritmo da função de verossimilhança alcançado ao final das iterações se aproxima, na maioria das execuções, de dois valores distintos, que se verificou serem -5620 e -6140, aproximadamente.

A Figura 4 também mostra que, para quantidades de iterações maiores ou iguais a 50, 100% das execuções do VTPSO resultaram em valores de LL superiores ou iguais a -6140. Resultados desta natureza podem mudar com os valores dos parâmetros do algoritmo, como sugere a investigação inicial apresentada na próxima subseção.

Efeito do valor do parâmetro c_3

Para avaliar a sensibilidade da distribuição de verossimilhança gerada pelo algoritmo VTPSO ao valor da constante c_3 foram feitos testes com 500 execuções em que os valores de c_3 foram escolhidos do conjunto $\{0,5 \;\; 1,0 \;\; 2,0 \;\; 3,0 \;\; 4,0\}$, sendo os valores de $(W \;\; c_1 \;\; c_2)$ fixados em $(0,7 \;\; 2,0 \;\; 2,0)$.

No que diz respeito ao número de iterações, foram considerados os valores de 10 e 20, obtendo-se as curvas de distribuição de verossimilhanças mostradas nas Figuras 5(a) e 5(b), respectivamente. Percebe-se nestas figuras que o aumento do valor de c_3 acarreta aumento do percentual das realizações com altos valores de verossimilhança, como resultado da maior influência do VT na atualização das posições das partículas. Além disso, comparando-se as duas figuras observa-se que com o aumento do número de iterações a dependência com o valor de c_3 diminui. Por exemplo, na Figura 5(b) quase não se percebe diferença nos valores de verossimilhança para valores de c_3 maiores ou iguais a $2,0$.

7. CONCLUSÃO

Neste artigo foi proposto e investigado o desempenho de um algoritmo híbrido (VTPSO) para estimação de máxima verossimilhança dos parâmetros de modelos HMM, baseado no treinamento de Viterbi e no algoritmo PSO.

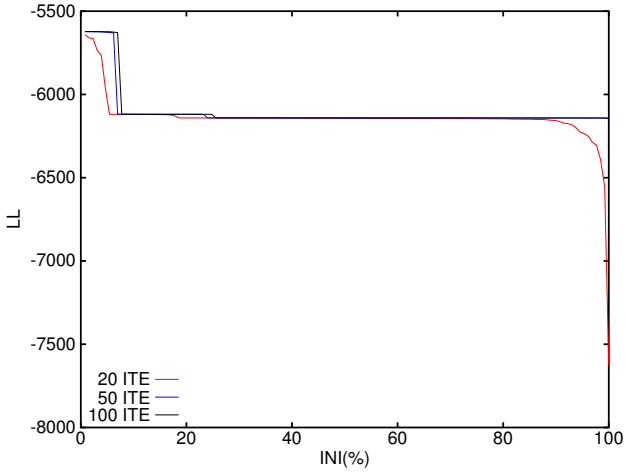


Figura 4: Distribuições de verossimilhanças do algoritmo VTPSO com diferentes valores do número de iterações.

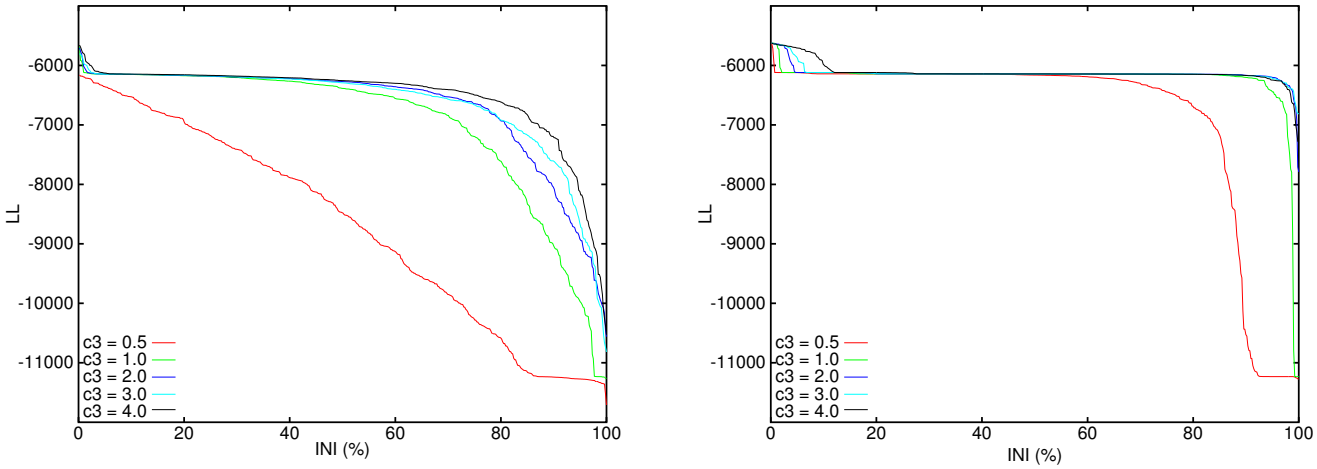


Figura 5: Distribuições de verossimilhanças do algoritmo VTPSO para diferentes valores do parâmetro c_3, com (a) 10 iterações e (b) 20 iterações.

Foi realizada uma comparação entre os desempenhos do algoritmo proposto e dos algoritmos PSO e BW na estimação de modelos HMM aplicados a canais com erros em surtos em que se constatou que o algoritmo VTPSO pode produzir resultados de verossimilhança significativamente melhores do que os outros dois, para idênticos tempos de processamento. Tais resultados indicam que o algoritmo proposto apresenta características vantajosas dos algoritmos PSO e VT.

Além disso, ficou bem ilustrada a importância do algoritmo VT para prover o algoritmo proposto de maior habilidade na maximização da função de verossimilhança. Cabe notar, no entanto, que deve ser feita uma análise mais detalhada dos valores dos parâmetros do VTPSO para se explorar adequadamente o seu potencial. Em trabalhos futuros pretende-se atacar em maior profundidade esta questão.

Pretende-se também investigar outras formas de aliar as vantagens dos algoritmos PSO e VT, visando obter algoritmos híbridos de complexidade menor que a do VTPSO. Uma alternativa para isso pode ser, por exemplo, a de restringir o emprego do VT, a cada iteração, a um subconjunte de partículas com melhores valores da função custo, atualizando as outras partículas segundo o algoritmo PSO convencional.

Referências

[1] J. Huang and K. Visweswariah. "Improved decision trees for multi-stream HMM-based audio-visual continuous speech recognition". In *IEEE Workshop on Automatic Speech Recognition Understanding*, 13 2009.

[2] Y. Zou, G. Shi, H. Shi and Y. Wang. "Image Sequences Based Traffic Incident Detection for Signaled Intersections Using HMM". In *Ninth International Conference on Hybrid Intelligent Systems*, volume 1, pp. 257 –261, 2009.

[3] G. Fink, S. Vajda, U. Bhattacharya, S. K. Parui and B. B. Chaudhuri. "Online Bangla Word Recognition Using Sub-Stroke Level Features and Hidden Markov Models". *International Conference on Frontiers in Handwriting Recognition (ICFHR)*, pp. 393–398, 2010.

[4] J. Garcia-Frias and P. Crespo. "Hidden Markov models for burst error characterization in indoor radio channels". *IEEE Transactions on Vehicular Technology*, vol. 46, no. 4, pp. 1006 –1020, nov 1997.

[5] L. Venkataramanan, R. Kuc and F. Sigworth. "Identification of Hidden Markov Models for Ion Channel Currents-Part III: Bandlimited Sampled Data". *IEEE transactions on signal processing*, vol. 48, pp. 376, February 2000.

[6] D. Hernandez, S. I. Marcus and P. J. Fard. "Analysis of a Risk-Sensitive Control Problem for Hidden Markov Chains". *IEEE transactions on automatic control*, vol. 44, pp. 1093, May 1999.

[7] K. Georgoulakis and S. Theodoridis. "Bind and semi-blind equalization using hidden Markov models and clustering techniques". *Signal processing*, vol. 80, pp. 1795, 2000.

[8] T. H. S. I. S. H. D. Umehara, M. Morikura. "Statistical impulse detection of in-vehicle power line noise using hidden Markov model". *IEEE International Symposium on Power Line Communications and Its Applications (ISPLC)*, pp. 341–346, 2010.

[9] X. Zhang, Y. Wang and Z. Zhao. "A Hybrid Speech Recognition Training Method for HMM Based on Genetic Algorithm and Baum Welch Algorithm". *Second International Conference on Innovative Computing*, 2007.

[10] A. Churbanov and S. Winters-Hilt. "Implementing EM and Viterbi algorithms for Hidden Markov Model in linear memory". *BMC Bioinformatics*, 2008.

[11] F. Yang, C. Zhang and G. Bai. "A Novel Genetic Algorithm Based on Tabu Search for HMM Optimization". In *Fourth International Conference on Natural Computation, 2008. ICNC '08.*, volume 4, pp. 57 –61, 2008.

[12] S. Phon-Amnuaisuk. "Estimating HMM Parameters Using Particle Swarm Optimisation". In *Proceedings of the EvoWorkshops 2009 on Applications of Evolutionary Computing*, 2009.

[13] L. Xue, J. Yin, Z. Ji and L. Jiang. "A Particle Swarm Optimization for Hidden Markov Model Training". In *8th International Conference on Signal Processing*, 16 2006.

[14] J. Lember and A. Koloydenko. "Adjusted Viterbi Training". *Journal Probability in the Engineering and Informational Sciences*, vol. 21, no. 3, July 2007.

[15] N. M. Paiva Junior, E. C. Marques and E. L. Pinto. "Viterbi Training for HMM Modelling of Burst Errors". *International Workshop on Telecommunications, IWT*, 2011.

[16] R. Eberhart and J. Kennedy. "A new optimizer using particle swarm theory". pp. 39 –43, oct 1995.

[17] M. Maca, D. NovÃ¡k and L. LhotskÃ¡. "Constraints in Particle Swarm Optimization of Hidden Markov Models". In *Intelligent Data Engineering and Automated Learning (IDEAL)*, volume 4224, pp. 1399–1406, 2006.

[18] M. Fernandes, E. Pinto and M. Grivet. "HMM Modelling of Burst Error Channels by Particle Swarm Optimization of the Likelihood Function". In *2010 International Telecommunications Symposium - ITS 2010*, volume 1, pp. 1 –5, 2010.

[19] N. Maciel, E. Marques and E. Pinto. "Desempenho do Algoritmo PSO na Estimacao ML de Modelos HMM para Erros em Surtos". *Simposio Brasileiro de Telecomunicacoes, SBrT'11*, 2011.

[20] J. Bilmes. "A Gentle Tutorial on the EM Algorithm Including Gaussian Mixtures and Baum-Welch". In *ICSI Technical Report TR-97-021*, 1997.

[21] E. Marques, N. Maciel and E. Pinto. "Tratamento de Restricoes Estocasticas no Algoritmo PSO para Estimacao ML de Modelos HMM". *Simposio Brasileiro de Telecomunicacoes, SBrT'11*, 2011.

[22] C.-X. Wang and W. Xu. "A New Class of Generative Models for Burst-Error Characterization in Digital Wireless Channels". *IEEE Transactions on Communications*, 2007.

[23] O. Salih, C.-X. Wang and D. Laurenson. "Three Layered Hidden Markov Models for Binary Digital Wireless Channels". In *IEEE International Conference on Communications, 2009. ICC '09.*, pp. 1 –5, 2009.

[24] "Sage reference manual". http://www.sagemath.org.