بنام‌خدا

# پروژهٔ درس پردازش گفتار

## مقایسهٔ روشهای بازشناسی بر مبنای GMM، VQ و DTW برای مصوت‌های فارسی

استاد : دکتر رزازی

تهیه : سامان پروانه

بهمن ۱۳۸۳

## ۱- مقدمه

بازشناسی گفتار با شناسایی یک لغت معین یا با شناسایی گوینده در ارتباط است. الگوریتمهای شناسایی کلمهٔ ایزوله [۱] تلاش می‌کنند تا کلمات ایزوله را تشخیص دهند که یکی از کاربردهای آن در سرویس تلفن اتوماتیک است. سیستمهای شناسایی اتوماتیکِ گفتار تلاش می‌کنند تا زبانهای تلفظ شدهٔ بصورت پیوسته [۲] را شناسایی کنند و حتی آنها را بوسیلهٔ یک پردازشگر کلمه به متن تبدیل کنند. این سیستمها اغلب از ویژگیهای گرامری استفاده می‌کنند تا صحت [۳] کارشان را افزایش دهند.

برای بازشناسی روشهای مختلفی وجود دارد که در طی این پروژهٔ عملی سه روش VQ، GMM و DTW پیاده‌سازی و مقایسه خواهند شد.

## ۲- ایجاد مجموعهٔ آموزشی برای VQ و GMM

اطلاعات مربوط به مصوتها در داخل فایل vowel.mat قرار دارد که از دو سوم اول دنباله‌های زمانی هر کدام از واجها برای منظور آموزش استفاده می‌شود و یک‌سوم باقیمانده جهت آزمون استفاده می‌شوند. پس در گام اول با توجه به مطلب فوق داده‌های تست و آموزش را از هم جدا می‌کنیم. تعداد داده‌های آموزشی و آزمون برای هر کدام از مصوتها در جدول ۱ آورده شده است.

جدول ۱ : تعداد داده‌های آموزشی و آزمون برای هر کدام از مصوتها

| | iy | ae | aa | ux | ow | eh | oy |
|---|---|---|---|---|---|---|---|
| تعداد دادهٔ آموزشی | ۴۷ | ۲۰ | ۲۰ | ۱۳ | ۱۴ | ۳۶ | ۵ |
| تعداد دادهٔ آزمون | ۲۴ | ۱۱ | ۱۰ | ۷ | ۸ | ۱۹ | ۳ |

در جدول فوق اعداد بر اساس تعداد مصوت خاصی است که توسط گوینده گفته شده است. همانطوریکه در قسمتهای بعدی خواهیم دید مصوت oy بدلیل اینکه دادهٔ کمی برای آموزش دارد برای ما مشکل‌ساز شده و باعث مسئلهٔ Over Fitting می‌شود.

---

۱ Isolated Word Recognition
۲ Continuous Spoken Language
۳ Accuracy

در داده‌های بالا از ۲۵ ویژگی استفاده شده است یعنی دادهٔ ما ۲۵ بعدی است.

### ۳- ایجاد VQ به روش Binary Splitting جهت بازشناسی مصوت

در روش Binary Splitting بتدریج و پس از بدست آوردن مراکز خوشه آنها را به قسمتهای کوچکتر تقسیم می‌کنیم. در این روش تعداد خوشه‌ها باید توانی از دو باشد. از آنجاییکه در این روش از الگوریتم k-means استفاده خواهیم کرد ابتدا توضیح مختصری دربارهٔ k-means داده خواهد شد سپس مسئلهٔ تقسیم باینری را ذکر خواهیم کرد. مراحل اصلی k-means عبارتند از:

۱- مقداردهی اولیه: ابتدا k مقدار اولیه برای مراکز خوشه در نظر می‌گیریم.

۲- داده‌های آموزشی را به نزدیکترین مرکز خوشه انتساب می‌دهیم.

$$\boldsymbol{x}\epsilon C_i, \quad d(\boldsymbol{x},\boldsymbol{\mu}_i) \le d(\boldsymbol{x},\boldsymbol{\mu}_j), \quad \forall i \ne j$$

۳- پس از انتساب در داخل هر خوشه مجدداً میانگین گیری کرده و عدد حاصله را بعنوان مرکز خوشه در نظر می‌گیریم در اصل مرحلهٔ بروزرسانی را داریم.

$$\boldsymbol{\mu}_i = E(\boldsymbol{x}), \quad \boldsymbol{x}\epsilon C_i, \quad 1 \le i \le K$$

۴- دوباره به مرحلهٔ دوم بر می‌گردیم تا شرط پایان فرا برسید.

مقدار دهی اولیه می‌تواند بر این اساس باشد که میانگینهای مراکز بطور تصادفی حول و حوش میانگین دادهٔ آموزشی توزیع می‌شوند که در روش اول پیاده‌سازی از این حالت استفاده شده است. حالت دیگر برای مقداردهی اولیه این است که میانگین داده را حساب کرده و یک عدد بسیار کوچک را از آن کم و زیاد می‌کنیم تا مراکز اولیه را برای خوشه‌ها بدست بیاوریم که در روش دوم از این حالت استفاده شده است.

از لحاظ شرط پایان ممکن است که یک تعداد مجاز برای تکرار در نظر بگیریم مثلاً ۲۰ تکرار و یا اینکه ممکن است تغییرات مراکز خوشه در دو تکرار پشت سرهم را اگر از یک حدی کوچکتر بود بعنوان شرط پایان در نظر بگیریم.که در روش اول برای پیاده‌سازی هر دو معیار پایان تعبیه شده است ولی در روش دوم فقط تعداد تکرار برای پایان الگوریتم در نظر گرفته شده است.

**توجه:** این الگوریتم به شدت به مقداردهی اولیه وابسته است و بنابراین بهتر است چند بار الگوریتم را اجرا کنیم تا مطمئن شویم در مینیمم محلی قرار نگرفته‌ایم.

در ضمیمهٔ ۱ متن برنامهٔ kmeans آورده شده است.

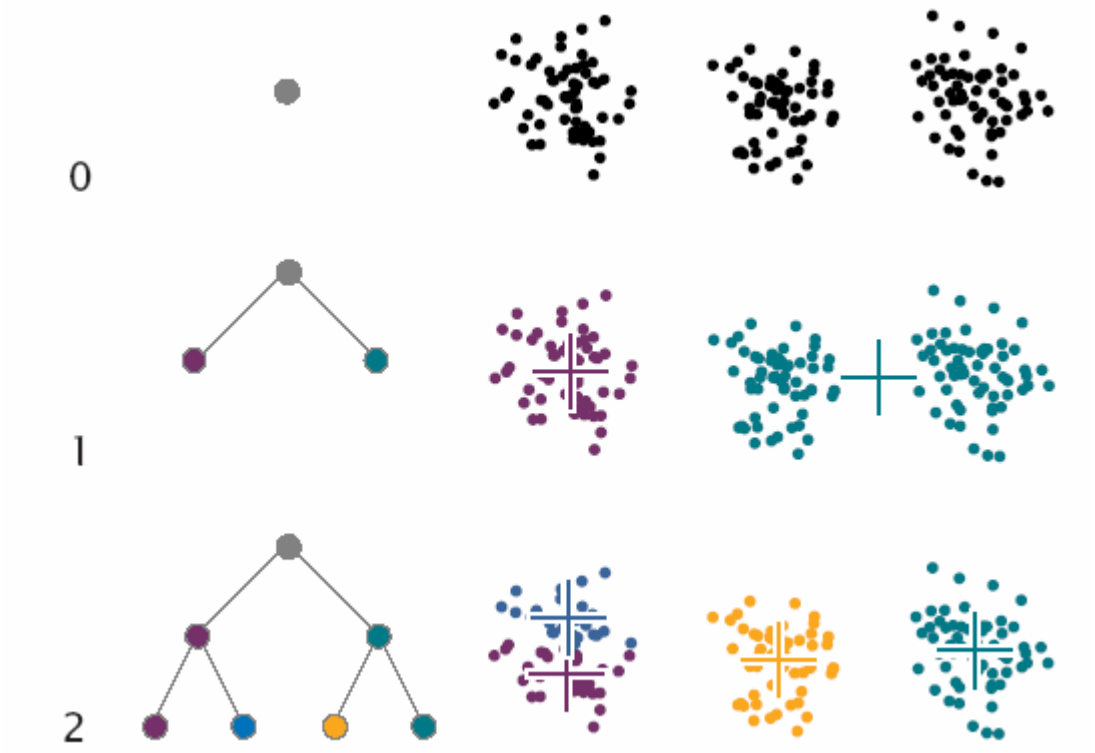
در Binary Splitting ابتدا دادهٔ ورودی به دو خوشه تقسیم می‌شود سپس مرکز هر کدام از خوشه‌ها مجدداً در حول و حوش میانگین داده‌های منتسب به آنها مقدار دهی اولیهٔ می‌شوند و هر کدام از خوشه‌ها مجدداً به دو مرکز دیگر تقسیم می‌شود و این فرآیند ادامه می‌یابد. پس ما در Binary Splitting بصورت

مکرر از kmeans استفاده می‌کنیم و در هر بار فراخوانی تابعی که kmeans را پیاده‌سازی می‌کند باید مراکز اولیهٔ خوشه‌ها را به آن ارسال کنیم. برای محاسبهٔ مقدار اولیهٔ مراکز خوشه‌ها می‌توانیم از رابطهٔ زیر کمک بگیریم:

$$\mu_i^+ = \mu_i(1+\epsilon)$$

$$\mu_i^- = \mu_i(1-\epsilon)$$

در شکل ۱ شمایی ظاهری از Binary Splitting برای ایجاد ۴ مرکز خوشه یعنی دو بار اجرای الگوریتم kmeans نشان داده شده است. متن برنامهٔ Binary Splitting در ضمیمهٔ ۲ آورده شده است.




شکل ۱ : شمایی ظاهری از Binary Splitting برای ایجاد ۴ مرکز خوشه

برای بکارگیری روش فوق در بازشناسی مصوت، ابتدا هر کدام از مصوتها را به تعداد مشخصی مرکز خوشه مثلاً ۴ خوشه VQ می‌کنیم . پس ما در این مثال خاص ما به ازای هر کدام از ۷ مصوت، ۴ مرکز خوشه داریم(۲۸ مرکز خوشهٔ کلی). در گام بعدی باید فاصلهٔ داده آزمون را از این مراکز خوشه بسنجیم. در این روش معیار ارزیابی الگوریتم به ازای دادهٔ آزمون مشخص می‌شود بدین صورت که فاصلهٔ هر کدام از قابهای دادهٔ آزمون را از تک‌تک مراکز خوشه می‌سنجیم و کمترین فاصله از مراکز خوشه را پیدا می‌کنیم و بسته به اینکه این فاصلهٔ مینیمم با کدام مرکز خوشهٔ مصوت حاصل شود، اندیس آنرا بعنوان

مصوت مشخص شده ذخیره می‌کنیم در گام بعدی و پس از اینکه مشخص کردیم هر کدام از قابها به مرکز خوشهٔ کدام مصوت نزدیک هستند عمل شمارش را انجام می‌دهیم و می‌بینیم قابها بیشتر به کدام مصوت منتسب شده‌اند و در گام آخر کل آن دادهٔ آزمون را بعنوان مصوتی که بیشتر تکرار شده است در نظر می‌گیریم.

پارامتر متغیر در این روش تعداد مراکز خوشه است که چون از Binary Splitting متقارن استفاده می‌کنیم باید توانی از دو باشد. در زیربخش بعدی اثر تغییر این پارامتر بررسی شده است.

## ٤- نتایج VQ به روش Binary Splitting جهت بازشناسی مصوت

نتایج این بخش به ازای سه بار تکرار الگوریتم و به ازای دو روشی که تفاوتهای آنها در بالا ذکر شد بدست آمده است و در جدول ۲ و ۳ آورده شده است. در جدول ۲ چون شرایط اولیه بصورت تصادفی تعیین شده است نتایج الگوریتم به ازای سه اجرای مختلف آورده شده است. چون شرایط اولیه در روش دوم به یک شکل تعیین می‌شود در نتیجه اجراهای مکرر برنامه به اعداد نزدیکی منجر می‌شود در نتیجه چون شرط پایان الگوریتم در این روش تعداد تکرار است، نتایج به ازای دو تعداد تکرار مختلف آورده شده است.

جدول ۲ : نرخ خطای متوسط برای روش ۱ و به ازای سه اجرای مختلف

| تعداد خوشه | ۲ | ۴ | ۸ | ۱۶ | ۳۲ | ۶۴ |
|---|---|---|---|---|---|---|
| اجرای اول | ٪۶۱,۳ | ٪۷۰,۶ | ٪۶۹,۵ | ٪۶۳,۹۸ | ٪۵۰ | ٪۴۹,۷ |
| اجرای دوم | ٪۵۵,۷۶ | ٪۶۵,۴۳ | ٪۵۷,۷۱ | ٪۵۲,۹۲ | ٪۴۵,۱۳ | ٪۵۵,۳۶ |
| اجرای سوم | ٪۵۴,۰۶ | ٪۶۸,۵۴ | ٪۷۰,۴ | ٪۵۴,۵۷ | ٪۵۱,۸۶ | ٪۴۷,۹۶ |
| میانگین | ٪۵۷,۰۴ | ٪۵۷,۰۴ | ٪۶۵,۸۷ | ٪۵۷,۱۵ | ٪۵۰ | ٪۵۱ |

جدول ۳ : نرخ خطای متوسط برای روش ۲ و به ازای دو مقدار مختلف برای تعداد تکرار مجاز الگوریتم

| تعداد خوشه | ۲ | ۴ | ۸ | ۱۶ | ۳۲ | ۶۴ |
|---|---|---|---|---|---|---|
| تعداد تکرار ۴۰ | ٪۵۸,۶ | ٪۵۹,۴۸ | ٪۷۵,۳۱ | ٪۴۵,۱۸ | ٪۴۵,۹ | ٪۴۴,۰۶ |
| تعداد تکرار ۶۰ | ٪۵۶,۵۹ | ٪۶۲,۶۳ | ٪۷۰,۵۸ | ٪۴۵,۱۳ | ٪۳۹,۱۴ | ٪۳۹,۵۵ |

همانطوریکه دیده می‌شود این روش بشدت تابع شرایط اولیه است و از طرفی نتایج خوبی را برای باظشناسی نداریم و درصد متوسط خطا بالا می‌باشد. از این روش بیشتر برای تخمین بدون ناظر اولیهٔ پارامترها استفاده می‌شود.

## ۵- استفاده از مدل مخلوط گوسی (GMM) جهت بازشناسی مصوت

در این روش تابع چگالی احتمال را بوسیلهٔ چند تابع گوسی مدل می‌کنیم. پس خط مشی کار بدین صورت است که هر کدام از مصوتها را بوسیلهٔ تابع گوسی ترکیبی مدل می‌کنیم که این مدل کردن شامل تعیین میانگین، واریانس و احتمال هر کدام از تابع گوسیهای منفرد است. تعیین این پارامترها توسط دادهٔ آموزشی برای هر کدام از مصوتها انجام می‌شود. پس در انتهای فاز آموزش (تخمین پارامتر) ما به ازای هر کدام از مصوتها یک مدل داریم. یعنی ۷ مدل مرجع ایجاد کرده‌ایم.

هر کدام از توابع گوسی دارای تابع چگالی احتمالی بفرم زیر می‌باشند:

$$p(\mathbf{x}\mid\omega_i)=\frac{1}{(2\pi)^{n/2}\left|\Sigma_i\right|^{1/2}}\exp\left[-\frac{1}{2}(\mathbf{x}-\boldsymbol{\mu}_i)^t\Sigma_i^{-1}(\mathbf{x}-\boldsymbol{\mu}_i)\right]$$

که از ترکیب آنها مدل GMM بدست می‌آید.

برای تعیین پارامترهای GMM از الگوریتم EM استفاده می‌کنیم. که بصورت خلاصه درزیر آورده شده است.

۱- تخمین اولیهٔ پارامترها: این مرحله معمولاً توسط روشهای بدون‌ناظر مثل kmeans انجام می‌شود.

۲- تابع کمکیِ لگاریتمِ همسانی را از روی داده بدست می‌آوریم.

۳- مقدار پارامترها را بگونه‌ای تعیین می‌کنیم که معادلهٔ کمکی ماکزیمم شود.

۴- پارامترهای تخمینی که در مرحلهٔ قبل بدشت آمد را در گام دوم جایگذاری می‌کنیم و فرآیند را تا زمان همگرایی ادامه می‌دهیم.

معمولاً در تکرار ۱۰ الی ۱۵ الگوریتم همگرا می‌شود. بروزرسانی میانگین، واریانس و احتمال ترکیب با استفاده از روابط زیر انجام می‌شود:

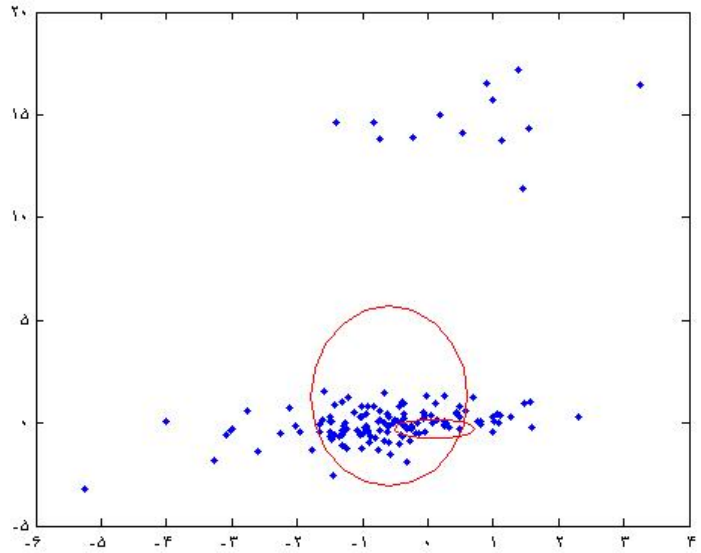
$$\hat{c}_k=\frac{\gamma_k}{\sum_{k=1}^{K}\gamma_k}=\frac{\gamma_k}{N}$$

$$\hat{\boldsymbol{\mu}}_k=\frac{\sum_{i=1}^{N}\gamma_k^i\mathbf{y}_i}{\sum_{i=1}^{N}\gamma_k^i}=\frac{\sum_{i=1}^{N}\frac{c_k p_k(\mathbf{y}_i\mid\boldsymbol{\Phi}_k)\mathbf{y}_i}{P(\mathbf{y}_i\mid\boldsymbol{\Phi})}}{\sum_{i=1}^{N}\frac{c_k p_k(\mathbf{y}_i\mid\boldsymbol{\Phi}_k)}{P(\mathbf{y}_i\mid\boldsymbol{\Phi})}}$$

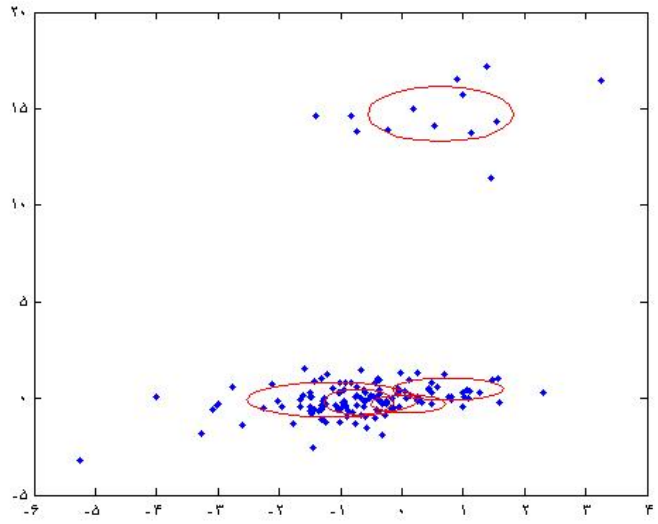
$$\hat{\Sigma}_k=\frac{\sum_{i=1}^{N}\gamma_k^i(\mathbf{y}_i-\boldsymbol{\mu}_k)(\mathbf{y}_i-\boldsymbol{\mu}_k)^t}{\sum_{i=1}^{N}\gamma_k^i}=\frac{\sum_{i=1}^{N}\frac{c_k p_k(\mathbf{y}_i\mid\boldsymbol{\Phi}_k)(\mathbf{y}_i-\boldsymbol{\mu}_k)(\mathbf{y}_i-\boldsymbol{\mu}_k)^t}{P(\mathbf{y}_i\mid\boldsymbol{\Phi})}}{\sum_{i=1}^{N}\frac{c_k p_k(\mathbf{y}_i\mid\boldsymbol{\Phi}_k)}{P(\mathbf{y}_i\mid\boldsymbol{\Phi})}}$$

در کارهای پردازش گفتار ماتریس کوواریانس را بصورت قطری در نظر می‌گیرند و سایر عناصر را صفر می‌کنند و بدین وسیله تعداد پارامترهای مجهول را کاهش می‌دهند. در مقالات مربوطه نشان داده شده است که ماتریس کوواریانس قطری نیز تخمین قابل قبولی را از تابع چگالی احتما داده در اختیارمان می‌گذارد.

هرچه تعداد مخلوطهای گوسی بیشتر شود تابع چگالی احتمال بصورت دقیقتری مدل می‌شود که این مطلب در شکل زیر نشان داده شده است.همانطوریکه در شکلها دیده می‌شود پنج مخلوط بصورت مناسبتری تابع چگالی احتمال را تخمین زده است.



شکل ۲ : استفاده از دو ترکیب برای مدل کردن



شکل ۳ : استفاده از پنج ترکیب برای مدل کردن

در ضمیمهٔ ۳ پیاده‌سازی تخمین پارامترهای GMM آورده شده است.

**مشکل پیاده‌سازی:** در هنگام پیاده‌سازی با مشکل underflow مواجه می‌شویم که این موضوع مخصوصاً برای حالتهایی که تعداد مخلوطها از ۵ عدد بیشتر است دیده می‌شود متاسفانه راه‌حلی برای حل این مشکل محاسباتی به ذهن من نرسید! ☹

پارامتر متغیر در این روش تعداد مخلوطها است که تا ۴ مخلوط تغییر کرده است و نتایج شبیه‌سازی در زیر بخش بعدی آورده شده است.

تا کنون برای هر کدام از مصوتها مدلی با تعداد ترکیب مشخص را بدست آوردیم. پس اکنون نوبت ارزیابی می‌باشد. برای ارزیابی می‌دانیم که هر دادهٔ آزمون از تعدادی قاب تشکیل شده است پس با توجه به توزیع احتمالی که در بالا بدست آوردیم احتمال اینکه هر مدلی، قاب اول را ایجاد کند بدست می‌آوریم همچنین احتمال اینکه مدل هر کدام از قابهای دیگر دادهٔ آزمون را ایجاد کند بدست می‌آوریم و در نهایت با فرض مستقل بودن داده‌ها آنها را در هم ضرب می‌کنیم. پس برای هر دادهٔ آزمون و به ازای ۷ مدلی که برای مصوتها داریم، عدد بدست می‌آید که از بین این اعداد بزرگترین را انتخاب کرده و دادهٔ آزمون را به مصوت متناظر با آن منتسب می‌کنیم.

### ۵- نتایج بازشناسی به روش GMM

در جدول ۴ نتایج بازشناسی به ازای تعداد ترکیبهای مختلف گوسی آورده شده است. ضرایب وزنی افقی، عمودی و قطری در این حالت برابر با یک هستند.

جدول ۴: درصد خطای متوسط به ازای سه اجرای مختلف GMM

| تعداد ترکیبهای گوسی | ۲ ترکیب | ۳ ترکیب | ۴ ترکیب | ۵ ترکیب |
|---|---|---|---|---|
| اجرای اول | ٪۴۵ | ٪۳۲ | ٪۳۹ | |
| اجرای دوم | ٪۴۲ | ٪۳۴ | ٪۳۷ | |
| اجرای سوم | ٪۴۰,۷ | ٪۴۲ | ٪۳۰,۴ | |
| میانگین | ٪۴۲,۵ | ٪۳۶ | ٪۳۵,۴ | ٪۳۶,۱ |

با توجه به جدول فوق دیده می‌شود که خطای این روش نسبتاً قابل قبول است و از طرفی دیده می‌شود که با زیاد شدن تعداد ترکیبهای گوسی درصد خطای متوسط کاهش می‌یابد که علت آن بهتر مدل شدن تابع چگالی احتمال مصوتها توسط GMM است.

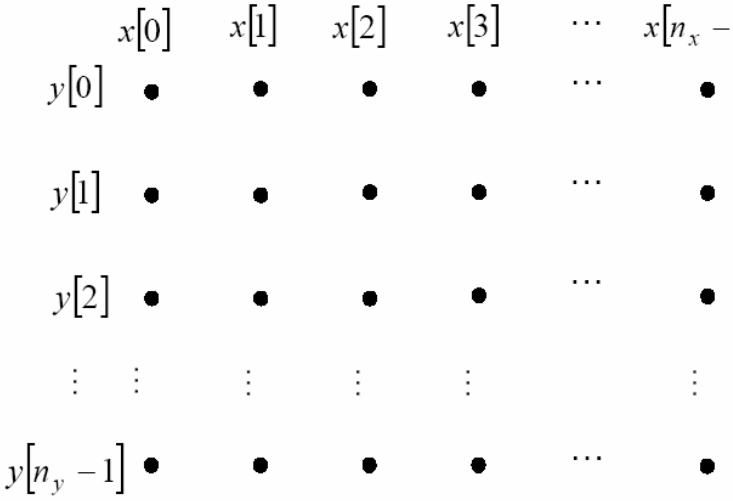
## ٦- روش DTW جهت بازشناسی مصوت

یکی از مشکلات اصلی در گفتار این است که حتی یک کلمهٔ خاص که توسط یک گوینده بیان می‌شود دارای طولهای مختلفی است یعنی بعبارت دیگر تعداد قابهای متفاوتی برای آن کلمه که دو بار تلفظ شده است وجود دارد. برا ی رفع این مشکل از برنامه‌ریزی دینامیکی استفاده می‌شود که در آن بر خلاف دو روش مطرح شدهٔ قبلی دادهٔ آموزشی به آن مفهوم وجود ندارد بلکه در اینجا یک سری Template مرجع داریم که سیگنالهای گفتار با طول متفاوت را نسبت به آن می‌سنجیم.

پارامتر متغیر در این روش تعداد Template های مرجع است که از ابتدای داده‌های مصوتها برداشته می‌شود مثلاً وقتی $M = ۲$ است یعنی از هر مصوت دو داده که می‌توانند دارای طولهای متفاوت و در نتیجه تعداد قابهای متفاوت باشند بعنوان Template مرجع برمی‌داریم پس در این حالت ۱۴ Template مرجع داریم که باید بقیهٔ مصوتها را نسبت به این مرجعها بسنجیم. حال سوال این است که یک داده‌ای که می‌خواهیم نوع آنرا مشخص کنیم چگونه به یک مصوت نسبت دهیم؟

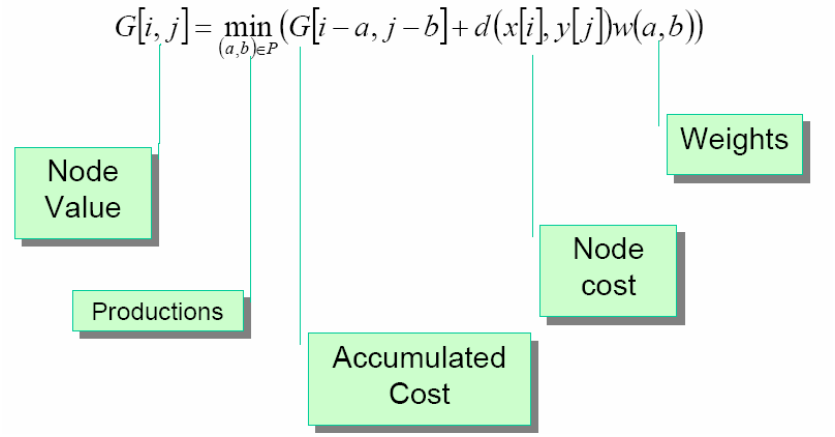
همانطور که کمی بعد توضیح داده می‌شود در روش DTW مفهوم فاصله را داریم پس در انتها به ازای هر سیگنال گفتار مورد بررسی و هر Template مرجع یک فاصله داریم، حال سیگنال را به مصوتی منتسب می‌کنیم که با Template مرجع متناظرِ آن کمترین فاصله را دارد.

در روش DTW با دو مفهوم فاصله روبرو هستیم یک فاصلهٔ محلی و دیگری فاصلهٔ کلی. در گام اول Template مرجع را روی محور y و سیگنال مورد بررسی را روی محور x در نظر می‌گیریم و به ازای تعداد قابهای آنها فضا را می‌شکنیم. این مطلب در شکل زیر آورده شده است.

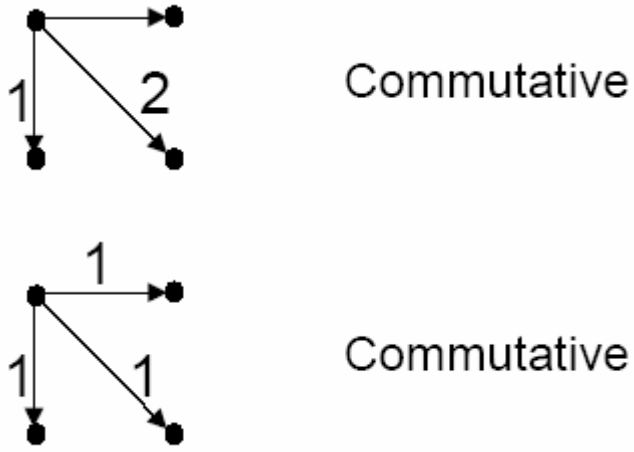



فضای کار DTW

در گام بعدی فاصلهٔ محلی را که بعنوان ارزش هر گره است ایجاد می‌کنیم. حال بصورت بازگشتی و با توجه به این مطلب که مسیری کوتاه است که تا قبل از این نقطه نیز کوتاه بوده باشد بدنبال کوتاهترین مسیر می‌گردیم و فاصلهٔ کلی را بر اساس رابطهٔ زیر بروزرسانی می‌کنیم.




که در رابطهٔ فوق d فاصلهٔ محلی است و G فاصلهٔ کلی می‌باشد. آرگومان min پیدا کردن مسیر کمینه را نشان می‌دهد و w ضریب وزنی انتقال از یک گره به گره دیگر است. برای این ضریب وزنی دو حالت متداول وجود دارد که در شکل زیر نشان داده شده‌اند.




شکل ۴

هر دو حالت فوق در داخل برنامه قابلیت پیاده‌سازی دارند. با توجه به مطالب فوق نقطهٔ پایین و سمت راست در انتهای الگوریتم کوتاهترین فاصلهٔ کلی را دارد پس از مقدار این گره برای مقایسهٔ مصوتهای مختلف استفاده می‌کنیم. در صورتیکه بخواهیم بغیر از بازشناسی از این روش استفاده کنیم برای مثال در alignment دو سیگنال گفتار باید مسیر برگشت را نیز ذخیره کنیم.

در simmx.m که بصورت تابع ایجاد شده است فاصلهٔ محلی محاسبه می‌شود و در dp.m الگوریتم dtw ایجاد می‌شود. این توابع در ضمیمهٔ ۴ آورده شده‌اند.

در این روش پارامتر متغیر تعداد Template های مرجع است و از آنجاییکه برای مصوت oy ، ۸ داده بیشتر در اختیار نداریم تعداد Template مرجع را ماکزیمم ۷ می‌گیریم که داده‌ای نیز برای آزمایش oy داشته باشیم.

## ۷- نتایج بازشناسی مصوت به روش DTW

در جدول ۵ نتایج بازشناسی به ازای تعداد الگوهای مرجع مختلف آورده شده است. به ازای الگوهای مرجع بیشتر میزان خطای متوسط کاهش یافته است که چنین اتفاقی قابل پیش‌بینی است.

جدول ۵ : درصد خطای متوسط به ازای تعداد الگوهای مختلف برای هر مصوت

| تعداد الگو | ۱ الگو | ۲ الگو | ۳ الگو | ۴ الگو | ۵ الگو | ۶ الگو | ۷ الگو |
|---|---|---|---|---|---|---|---|
| درصد خطا | ٪۵۸,۷۴ | ٪۵۸,۸ | ٪۶۰,۵ | ٪۵۷,۴ | ٪۵۰,۲ | ٪۴۲,۰۷ | ٪۴۳,۱۸ |