

## مدل‌های مبتنی بر یادگیری عمیق در شناسایی سیستم‌های غیر خطی

وحید محمدزاده ایوقی<sup>۱</sup>، مهدی علیاری شوره‌دلی<sup>۲</sup>

<sup>۱</sup> دانشجوی دکتری مهندسی کنترل، دانشگاه صنعتی خواجه نصیرالدین طوسی، تهران، ایران vmohammadzadeh@email.kntu.ac.ir

<sup>۲</sup> دانشیار گروه مکترونیک، دانشکده مهندسی برق، دانشگاه صنعتی خواجه نصیرالدین طوسی، تهران، ایران aliyari@kntu.ac.ir

پذیرش: ۱۴۰۲/۰۶/۲۰

دریافت: ۱۴۰۲/۰۵/۰۵

**چکیده** – مدل‌های مبتنی بر یادگیری عمیق عملکرد بسیار مناسبی در مدل‌سازی مسائل پیچیده در بینایی ماشین و پردازش زبان طبیعی از خود نشان داده‌اند که این موضوع به ماهیت غیر خطی و فراپارامتری این مدل‌ها نسبت داده می‌شود. روش‌های شناسایی سیستم‌های غیر خطی، می‌تواند از ابزارهای توسعه یافته در حوزه یادگیری عمیق بهره‌مند شوند که این امر موجب گسترده شدن ابزارهای موجود برای انتخاب یک مدل مناسب خواهد شد. از این رو، در این مقاله قصد داریم تا روش‌ها و ساختارهای موجود در یادگیری عمیق را از دیدگاه شناسایی سیستم‌های غیر خطی مرور کنیم. اگرچه مرور نسبتاً جامعی از ابزارهای قابل استفاده در حوزه شناسایی سیستم‌های غیر خطی ارائه خواهد شد، اما تمرکز اصلی این پژوهش بر کاربرد مدل‌های متغیر پنهان در شناسایی فضای حالت غیر خطی است. مدل‌های متغیر پنهان دسته‌ای از مدل‌های یادگیری عمیق هستند که در گروه مدل‌های مولد قرار می‌گیرند. نسخه اصلی این مدل‌ها، تنها قابلیت تولید داده‌های ایستا را داراست. اما با ترکیب شبکه‌های عصبی بازگشتی و خود رمزنگار تغییراتی، قابلیت تولید داده‌های ترتیبی برای این مدل‌ها نیز فراهم شده است. همچنین نسخه ساختاریافته‌ای از این مدل‌ها نیز که منطبق بر سیستم‌های کنترل است، ارائه خواهد شد. مطالعه انجام شده نشان می‌دهد که عملکرد این مدل‌ها با مدل‌های پیشین و کلاسیک موجود، قابل قیاس است.

**کلیدواژه:** مدل‌های یادگیری عمیق، شناسایی سیستم‌های غیر خطی، مدل‌های متغیر پنهان، خود رمزنگار تغییراتی

## Deep Learning based Models for Nonlinear System Identification

Vahid Mohammadzadeh Ayooghi and Mahdi Aliyari-Shoorehdeli

**Abstract** – Deep learning-based models appropriately perform in modeling complex problems in computer vision and natural language processing (NLP). With this in mind, nonlinear system identification methods can benefit from tools developed in deep learning, leading to enriched frameworks to choose from. For this purpose, we are going to review some potential structures and methods of deep learning that can be used in nonlinear system identification. Although we comprehensively review the applicable tools of deep learning to system identification, this paper mainly focuses on using latent variable models (LVM) for identifying nonlinear state space models. LVMs are powerful tools for extending generative models primarily developed for only generating static data, yet by a combination of recurrent neural network (RNN) and variational auto-encoders (VAE), they can also generate sequential data. A structured version of introduced models compatible with control systems will also be given. The study shows that the deep learning models have a comparative performance to traditional and classic ones.

**Keywords:** Deep learning, nonlinear system identification, latent variable models, VA

## ۱- مقدمه

از دیدگاه ریاضی، مسئله شناسایی سیستم، مسئله پیدا کردن یک تابع ریاضی است که ورودی های این تابع ریاضی، متغیرهای دینامیکی ورودی، خروجی و یا خطای پیش بینی می تواند باشد. روش های شناسایی خطی، خروجی سیستم را در هر لحظه بر اساس ترکیب خطی ای از این ورودی ها بیان می کنند درحالی که روش های شناسایی غیرخطی، خروجی را در هر لحظه بر اساس ترکیب خطی ای از توابع غیرخطی که بر روی این ورودی ها اعمال می شوند، محاسبه می کنند که این رویکرد با نام مدل توابع پایه<sup>۲۱</sup> شناخته شده است [۳]. از دیدگاه جبر خطی، می توان گفت هر دو روش سعی در پیدا کردن یک فضای برداری دارند که در آن خروجی به سهولت ساخته می شود. برای روش های خطی این فضای مربوط به زیر فضای ورودی های مدل است درحالی که روش های غیرخطی فضایی متفاوت از فضای ورودی را به عنوان فضای حل مسئله انتخاب می کنند. در واقع دسته بندی معرفی شده برای روش های شناسایی غیرخطی، تفاوت دیدگاه های موجود برای مدل سازی این نگاشت غیرخطی عنوان می کنند. به عنوان مثال برای مدل های جعبه سیاه، نوع نگاشت غیرخطی محاسبه شده اهمیت چندانی ندارد و تنها مسئله مهم نزدیک شدن خروجی مدل به خروجی سیستم است. یا در روش های مبتنی بر کرنل، در دید کلی، فرایندهای گوسی، این نگاشت غیرخطی بر اساس شباهت بین نمونه ها در فضای ورودی ساخته می شود [۳]، [۵]، [۱۰] و [۱۱].

دیدگاه بیان شده برای روش های شناسایی سیستم ها، دقیقاً منطبق بر فرایندی است که در حوزه یادگیری ماشین<sup>۲۲</sup> برای حل یک مسئله داده محور انجام می شود. داده هایی که در اختیار طراح برای حل یک مسئله مشخص قرار می گیرد، ابتدا به یک فضای نگاشت می شوند که در فضای جدید حل مسئله سادگی قابل انجام باشد. به نمایش داده ها در فضای جدید، در ادبیات یادگیری ماشین، بازنمایی<sup>۲۳</sup> گفته می شود. تمام تلاش ها در یک مسئله یادگیری ماشین یافتن یک بازنمایی مناسب برای حل مسئله است [۱۲]. در یادگیری ماشین کلاسیک، این بازنمایی بر اساس مجموعه ای از نگاشت های ثابت اعم از چند جمله ای<sup>۲۴</sup>، توابع پایه شعاعی<sup>۲۵</sup>، توابع کرنل<sup>۲۶</sup>،

شناسایی سیستم ها<sup>۱</sup> یکی از زمینه های حوزه کنترل خودکار<sup>۲</sup> است که بر اساس داده های ورودی و خروجی جمع آوری شده از یک سیستم واقعی، تلاش در پیدا کردن یک مدل مناسب دینامیکی دارد [۱]. اهمیت این مسئله و توسعه ابزارهای آن در یک چارچوب مشخص، برای اولین بار در [۲] برای سیستم خطی تغییر ناپذیر با زمان<sup>۳</sup> آغاز شد و به مرور زمان منجر به توسعه ابزارهای متنوع حوزه شناسایی سیستم های خطی شده است [۱]، [۳] و [۴]. محدودیت این روش ها در مدل کردن فرآیندهای صنعتی و واقعی که عمدتاً دارای یک رفتار غیرخطی<sup>۴</sup> هستند [۵] و [۳]، پژوهشگران را بر آن داشت تا یک چارچوب برای شناسایی سیستم های غیرخطی نیز ایجاد شود که نمونه مناسبی از مجموعه این روش ها در [۳] به تفصیل آمده است. فارغ از عدم موفقیت تلاش های روزافزون، روش های توسعه داده شده برای مدل سازی غیرخطی، به طور کلی به سه دسته توسعه تابعی<sup>۵</sup>، بلوک-محور<sup>۶</sup> و جعبه سیاه<sup>۷</sup> دسته بندی می شوند [۶]. روش توسعه ولترا<sup>۸</sup> و وینر<sup>۹</sup> از جمله رویکردهای مشهور توسعه تابعی است [۶]، [۷] و [۸]. این روش ها مسئله شناسایی سیستم را، به صورت یافتن یک تابعی<sup>۱۰</sup> از ورودی های سیستم به خروجی های سیستم مشاهده می کنند، و این تابعی را به شیوه های خاص خود تجزیه می کنند. بر این اساس مسئله شناسایی سیستم، به مسئله تخمین پارامترهای هر بخش تجزیه شده تبدیل خواهد شد [۶]. رویکرد بلوک-محور، با فرض جداپذیر بودن فرآیند<sup>۱۱</sup>، یک سیستم غیرخطی را به یک سیستم خطی دینامیکی و یک سیستم غیرخطی ایستاک<sup>۱۲</sup> تجزیه می کنند که در آن سیستم غیرخطی هر نوع تقریب زنی اعم از شبکه های عصبی<sup>۱۳</sup> و سیستم های فازی<sup>۱۴</sup> می تواند باشد. بر اساس اتصال این دو بخش تجزیه شده، مدل های متفاوتی حاصل می شود که مدل های وینر<sup>۱۵</sup>، همراشتاین<sup>۱۶</sup> و وینر-همراشتاین<sup>۱۷</sup> از جمله معروف ترین مدل های این رویکرد است [۹]. رویکرد جعبه سیاه مجموعه ای از دینامیک های ورودی را دریافت می کند و نگاشتی از آنها را بر روی مجموعه ای از داده های خروجی محاسبه می کنند. روش های مبتنی بر کرنل<sup>۱۸</sup>، مدل های خطی محلی<sup>۱۹</sup>، و استفاده از شبکه های عصبی بازگشتی<sup>۲۰</sup> از جمله این روش ها هستند [۳].

<sup>14</sup> Fuzzy systems<sup>15</sup> Wiener model<sup>16</sup> Hammerstein model<sup>17</sup> Wiener-Hammerstein model<sup>18</sup> Kernel-based methods<sup>19</sup> Locally linear models<sup>20</sup> Recurrent neural networks<sup>21</sup> Basis function models<sup>22</sup> Machine learning<sup>23</sup> Representation<sup>24</sup> Polynomial mapping<sup>25</sup> Radial basis function<sup>26</sup> Kernel functions<sup>1</sup> System identification<sup>2</sup> Automatic control<sup>3</sup> Linear time-invariant system<sup>4</sup> Nonlinear behavior<sup>5</sup> Functional expansion<sup>6</sup> Block-oriented models<sup>7</sup> Black box<sup>8</sup> Volterra expansion<sup>9</sup> Wiener expansion<sup>10</sup> Functional<sup>11</sup> Process separability<sup>12</sup> Static nonlinearity<sup>13</sup> Neural networks

برگشت پذیری به فضای اصلی متغیرهای ورودی را داراست. بنابراین مسئله کنترل بازنمایی در این رویکرد منجر به یادگیری فضایی خواهد شد که مشخصات ثابت و معین متغیرهای را ورودی را داشته باشد. با استناد به این اصل که در متغیرهای یک مسئله پیچیده مانند تشخیص گفتار، هر دو مشخصات تصادفی و معین وجود دارد، این رویکرد قابلیت استخراج تمام مشخصات مرتبط با یک متغیر را ندارد و تنها مشخصات ثابت را استخراج می کند [۱۷]، [۱۸] و [۱۹]. مدل های احتمالاتی، متغیرهای مربوط به یک ساختار را به دو بخش تقسیم می کنند: متغیرهای مشاهده شده<sup>۱۴</sup> و متغیرهای پنهان<sup>۱۵</sup> [۱۸]. این دسته از مدل ها تحت عنوان مدل های متغیر پنهان<sup>۱۶</sup> نیز شناخته می شوند و نوع بازنمایی آموزش دیده شده توسط این مدل ها با رویکرد نگاشت مستقیم تفاوت های زیادی دارد. این مدل ها بر این فرض استوار هستند که متغیرهای یک مسئله دارای دسته ای از ویژگی های پنهان هستند که این ویژگی ها تغییرات داده ها را توصیف می کنند. مسئله یادگیری بازنمایی در این مدل ها به صورت یک نگاشت تصادفی از متغیرهای پنهان به متغیرهای مشاهده شده تعریف می گردد. این مدل ها توانایی استخراج مشخصات تصادفی در متغیرهای ورودی را دارا هستند. ماشین بولتزمن محدود شده<sup>۱۷</sup> [۲۰]، [۲۱] و [۲۳] و خودرمننگار تغییراتی از جمله این روش ها هستند. ساختار خودرمننگار تغییراتی دقیقاً با ساختار خودرمننگار عادی در رویکرد مستقیم یکسان است. بنابراین امکان استخراج مشخصات ثابت و تصادفی به صورت توامان توسط این ساختار وجود ندارد. مدل هایی در حوزه یادگیری ماشین توسعه داده شده اند که با ترکیب دو نگاشت معین و تصادفی هر دو مشخصات را از یک دسته ورودی محاسبه می کنند. به عنوان مثال در [۱۷] شبکه عصبی بازگشتی در کنار خودرمننگار تغییراتی، امکان استخراج هر دو مجموعه از ویژگی ها را دارد به این صورت که شبکه عصبی بازگشتی ویژگی های ثابت و وابسته به زمان، و خودرمننگار تغییراتی مشخصات تصادفی را استخراج می کند. روش یادگیری رویه، به طور مشابه با متغیرهای پنهان عمل می کند با این تفاوت که یادگیری بازنمایی بر اساس نزدیکی و شباهت بین نقاطی از داده های ورودی است که مدل مجبور به یادگیری آنها خواهد شد. یادگیری متضاد<sup>۱۸</sup> [۲۲] از جمله روش هایی است که بر اساس شباهت بین نمونه های مجموعه ای از داده گان، بازنمایی مناسبی را از داده گان آموزش می بینند. اگرچه بسته به نوع کاربردها و مسائل متفاوت، هر یک از سه روش

تبدیلات حوزه فرکانس<sup>۱</sup> انجام می شود که به دلیل شکل و محدودیتی که در اعمال آنها به مجموعه داده های ورودی دارد [۱۲] و [۱۳]، انتخاب و استفاده از آنها تا حد زیادی به دانش طراح بستگی دارد. مهم تر اینکه این نگاشت های ثابت در برخی از شرایط، مانند داده های با بعد بالا<sup>۲</sup> و سیستم ها پیچیده<sup>۳</sup>، عملکرد مناسبی از خود نشان نمی دهند و به بیان بهتر، در مواجهه با برخی از مسائل نیاز است تا بازنمایی های پیچیده تری از متغیرهای ورودی مسئله پیدا کرد که طبیعتاً امکان این مسئله به صورت دستی وجود ندارد. زیرا تعداد نامتناهی تابع وجود دارد که می تواند به داده ها اعمال شود. این مشکل دقیقاً فلسفه ایجاد مدل هایی موسوم به مدل های عمیق<sup>۴</sup> است که نسخه معین<sup>۵</sup> آن شبکه های عصبی عمیق<sup>۶</sup> است [۱۳]. شبکه های عصبی عمیق نوعی از مدل های عمیق هستند که بر روی داده های ورودی نگاشت های غیرخطی پی در پی اعمال می کند که در نتیجه اعمال این نگاشت های غیرخطی، نگاشت پیچیده تری از متغیرهای ورودی ساخته می شود که با تکیه بر تکنولوژی های یادگیری که روز به روز در حال پیشرفت هست، امکان کنترل این بازنمایی نیز وجود دارد [۱۳]. کنترل بازنمایی به معنای شفافیت در یادگیری لایه های پنهان یک شبکه عصبی است که این موضوع درک مدل را از داده ها نشان می دهد.

الگوریتم های متفاوتی برای ایجاد درک مناسب از داده ها، و به بیان بهتر کنترل بازنمایی، در حوزه یادگیری ماشین ایجاد شده اند که عمدتاً جزو دسته یادگیری بدون نظارت هستند [۱۴]. این روش ها در حالت کلی به سه دسته قابل تقسیم هستند، نگاشت مستقیم<sup>۷</sup>، مدل های احتمالاتی<sup>۸</sup>، و یادگیری رویه<sup>۹</sup> [۱۵]. روش نگاشت مستقیم مبتنی بر یادگیری بازنمایی ای از داده ها است که این بازنمایی اولاً از یک نگاشت معین حاصل می شود و ثانیاً این بازنمایی توانایی بازگشت به فضای اصلی را دارد. این روش های به صورت مستقیم از متغیرهای ورودی، ویژگی هایی را در فضایی با بعد پایین تر استخراج می کنند. مهم ترین ابزاری که در این دسته بندی قرار می گیرد خودرمننگارها<sup>۱۱</sup> هستند که از دو شبکه عصبی کدکننده<sup>۱۱</sup> و کدگشا<sup>۱۲</sup> تشکیل می شوند و شبکه با حداقل کردن خطای بازایی آموزش می بیند [۱۳] و [۱۵]. خودرمننگارها انواع متفاوتی دارند که تمام انواع به جز خودرمننگار تغییراتی<sup>۱۳</sup> [۱۶]، مثال هایی از رویکرد نگاشت مستقیم برای یادگیری بازنمایی هستند [۱۵]. رویکرد نگاشت مستقیم، از طریق یک نگاشت معین، بازنمایی ای را آموزش می بیند که این بازنمایی قابلیت

<sup>11</sup> Encoder

<sup>12</sup> Decoder

<sup>13</sup> Variational auto-encoder

<sup>14</sup> Observed variables

<sup>15</sup> Latent variables

<sup>16</sup> Latent variable models

<sup>17</sup> Restricted Boltzmann machine

<sup>18</sup> Contrastive learning

<sup>1</sup> Frequency domain transformation

<sup>2</sup> High dimensional data

<sup>3</sup> Complex systems

<sup>4</sup> Deep models

<sup>5</sup> Deterministic version

<sup>6</sup> Deep neural network

<sup>7</sup> Direct mapping

<sup>8</sup> Probabilistic models

<sup>9</sup> Manifold learning

<sup>10</sup> Auto-encoders

یاد شده می تواند برای یادگیری بازنمایی مناسب باشد، اما یادگیری بازنمایی به کمک مدل های احتمالاتی از اهمیت بالایی برخوردار است. بدلیل ماهیت نگاشتی که این مدل ها استفاده می کنند، امکان تعمیم دهی<sup>۱</sup> مدل به شرایط نادیده در این مدل ها بالاتر است زیرا این مدل ها تغییرات و عدم قطعیت ها موجود در مجموعه ای از داده ها را مدل می کنند. به بیان بهتر، این مدل ها قابلیت مدل کردن توزیع احتمال توام بین متغیرهای مشاهده شده و متغیرهای پنهان را داراست. در ادبیات یادگیری ماشین، این نوع مدل ها موسوم به مدل های مولد<sup>۲</sup> است [۱۳] و [۱۶]. مدل های مولد به هدف های متفاوتی توسعه پیدا می کنند که یکی از این اهداف بهبود بازنمایی و بالا بردن ظرفیت تعمیم دهی مدل های یادگیری ماشین است [۱۳].

اگرچه زمینه شناسایی سیستم ها به لحاظ روش های موجود غنی است، اما استفاده از شبکه های عصبی عمیق به منظور شناسایی سیستم ها مزیت های فراوانی مانند حل مشکل بالابودن بعد متغیرهای ورودی و مدل سازی سیستم های پیچیده را به دنبال خود دارد. همچنین استفاده از تکنولوژی ها و الگوریتم های آموزشی در این حوزه، نه تنها روش های شناسایی سیستم ها را توسعه می دهد بلکه می تواند موجب به بهبود آنها شده و در برخی موارد تفسیرپذیری و شفافیت مدل های جعبه سیاه را تضمین کند. در این مقاله قصد داریم، تا ابزارهای یادگیری عمیق که در حوزه شناسایی سیستم ها مورد استفاده قرار گرفته اند، مورد بررسی و توسعه قرار دهیم. با توجه به اهمیت مدل های احتمالاتی برای یادگیری بازنمایی، این مدل ها در شناسایی ورودی - خروجی و شناسایی فضای حالت می توانند منفعت های زیادی برای جامعه مهندسی کنترل ایجاد کند. تمرکز اصلی این پژوهش بر روی مدل های متغیر پنهان است که امکان شناسایی فضای حالت برای سیستم های غیرخطی را فراهم می کند.

ظرفیت های استفاده از یادگیری عمیق در حوزه شناسایی سیستم ها مسئله جدیدی نیست و ارتباط بین دو حوزه شناسایی سیستم و یادگیری عمیق در مقاله مروری [۲۳] و مقاله [۲۴] آورده شده است. در [۲۴] و [۲۵] از شبکه های عصبی عمیق و خاصیت یادگیری ویژگی سلسله مراتبی آنها استفاده شده است و مدل های عمیق NARX<sup>۳</sup> و NFIR<sup>۴</sup> توسعه داده شده اند. در عمده روش های شناسایی سیستم، فرض بر این است که متغیرهای خروجی از یک توزیع نرمال پیروی می کنند حال آنکه ممکن است در برخی کاربردها این فرض نادرست باشد. در [۲۶] با استفاده از شبکه های عصبی عمیق و مدل های مبتنی بر انرژی<sup>۵</sup> [۲۷]، مسئله شناسایی

سیستم را برای یک حالت کلی که در آن متغیرهای خروجی می توانند از هر توزیع احتمالی پیروی کنند، بیان شده است. چون ترکیب شبکه های عصبی عمیق و مدل های مبتنی بر انرژی می توانند هر نوع توزیع دلخواهی را تقریب بزنند، این رویکرد برای هر توزیعی از متغیرهای خروجی عملکرد خوبی خواهد داشت [۱۳]. در [۲۸] از ساختار شبکه های عصبی پیچشی زمانی<sup>۶</sup> برای شناسایی سیستم ها استفاده شده است. شبکه عصبی پیچشی زمانی همانند شبکه کانولوشنالی معمولی عمل می کند با این تفاوت که به منظور حفظ گام های زمانی ورودی و خروجی هر لایه از آن باید دارای بعد برابر باشند و عملگر کانولوشن باید علی<sup>۷</sup> باشد [۲۹]. استفاده از شبکه های کانولوشنالی زمانی می تواند در دسته روش های شناسایی بلوک - محور باشد چرا که به دلیل اعمال عملگر کانولوشن علی، می توان متصور شد که دنباله داده ورودی ابتدا از یک فیلتر خطی عبور می کند و سپس یک عملگر غیرخطی، تابع سیگموئید<sup>۸</sup> و یا یک سوساز خطی<sup>۹</sup>، بر روی آن اعمال می شود. چون برای هر شبکه چندین فیلتر در نظر گرفته می شود، می توان گفت فرایند فیلتر کردن خطی و اعمال تابع غیرخطی چندین بار صورت می پذیرد و خروجی ها با هم تجمع می شوند تا خروجی مدل را تخمین بزنند. این فرایند مشابه مدل موازی وینر<sup>۱۰</sup> است [۹]. در [۳۰] و [۳۱] از مدل های عمیق مبتنی بر کرنل<sup>۱۱</sup> برای شناسایی سیستم های غیرخطی استفاده شده است. روش های مبتنی بر کرنل خروجی را بر اساس شباهتی که بین نمونه های متفاوت در فضای ورودی وجود دارند، خروجی های متناظر با هر ورودی با شباهت به دست آمده ترکیب خطی می کنند. در مدل های عمیق مبتنی بر کرنل، تابع شباهت سنج با استفاده از شبکه های عصبی عمیق به وجود آمده است. در [۳۲] از رویکردی مشابه با خودرمنگاراها استفاده شده است و فضای حالت سیستم های خطی تغییر ناپذیر با زمان شناسایی شده است. در [۲۴] و [۳۳] و [۳۴] و [۳۵] با استفاده از ترکیب شبکه های عصبی عمیق و مدل های فضای حالت، مدلی موسوم به فضای حالت عمیق برای سیستم های غیرخطی شناسایی شده است. در [۳۶] با استفاده از یک شبکه عصبی بازگشتی پارامترهای یک مدل خطی فضای حالت تغییر با زمان<sup>۱۲</sup> را تخمین می زند. بر اساس این رویکرد مدل انتقال حالت و مدل مشاهدات هر دو از یک مدل خطی تغییرپذیر با زمان تبعیت می کنند اما چون نیاز است داده های زیادی از یک رویداد زمانی در دست باشد تا پارامترهای متغیر با زمان را تقریب بزنند، از یک شبکه عصبی بازگشتی برای تقریب پارامترها استفاده می کند. فلسفه استفاده از آن به این دلیل است که اگر پارامترها مدل با زمان تغییر کند اثر این تغییرات در

<sup>7</sup> Causal<sup>8</sup> Sigmoid function<sup>9</sup> Rectified linear units<sup>10</sup> Parallel wiener model<sup>11</sup> Deep kernel-based models<sup>12</sup> Linear time-variant system<sup>1</sup> Generalization<sup>2</sup> Generative models<sup>3</sup> Deep nonlinear ARX<sup>4</sup> Deep nonlinear FIR<sup>5</sup> Energy-based models<sup>6</sup> Temporal convolutional networks (TCN)

بسیار کم تر قابلیت تقریب زده شدن دارد. روش های متعددی برای توسعه این رویکرد وجود دارد که در [۵۰] خلاصه شده اند. اگرچه پژوهش های انجام شده به منظور فشرده سازی مدل، قابلیت استفاده برای مدل های توسعه داده شده در حوزه شناسایی سیستم ها را نیز دارد، اما به کارگیری مدل های عمیق در حوزه شناسایی سیستم ها نیاز است تا با دقت بیش تری صورت گیرد و این چالش مهم پاسخ داده شود. زیرا حوزه هایی مانند حوزه مهندسی کنترل، در بسیاری از کاربردها مانند طراحی کنترلر، نیازمند شفافیت بیش تری در فرآیند مدل سازی هستند و زمان استنتاج در این کاربردها از اهمیت بالایی برخوردار است.

تمرکز اصلی این مقاله بر روی استفاده از مدل های متغیر پنهان به منظور شناسایی سیستم های غیرخطی است. در بخش ۲ ادبیات مربوط به حوزه یادگیری عمیق و ابزارهای مورد استفاده برای مدل سازی سیستم های دینامیکی ارائه خواهد شد و سپس در بخش ۳ مدل های متغیر پنهانی که در حوزه شناسایی سیستم ها ورود کرده اند مورد بررسی قرار می گیرد. در بخش ۴، نسخه تغییر یافته ای از مدل های متغیر پنهان که دارای ساختار هستند است ارائه می شود، و نهایتاً موثر بودن مدل معرفی شده از طریق آزمایش بر روی سه نوع سیستم معیار<sup>۱۰</sup> غیرخطی، به نمایش گذاشته خواهد شد.

## ۲- ابزارهای یادگیری عمیق برای مدل سازی

### سیستم های دینامیکی

یکی از مسائل مهم یادگیری ماشین، تخمین تابع چگالی احتمال توام<sup>۱۱</sup> بر روی مجموعه ای از متغیرهای به هم مرتبط<sup>۱۲</sup> است. مسئله استنتاج<sup>۱۳</sup> و یادگیری توزیع توام بر روی این متغیرها بدلیل ناتوانی در کشف همبستگی بین متغیرها به صورت مستقیم، مسئله بسیار دشواری است که معمولاً برای حل آن از مدل های متغیر پنهان<sup>۱۴</sup> استفاده می شود که در آن فرض می شود، یک مکانیزم پنهان وجود دارد که می تواند تغییرات موجود در متغیرها را مدل کند. این مکانیزم پنهان به صورت لایه ای از متغیرهای تصادفی مدل می شود که دارای توزیع از پیش مشخص  $p(z)$  است که هر نمونه از آن توسط مدل  $p(x|z)$  توانایی تولید هر نمونه  $x$  را خواهد داشت. فرض می کنیم متغیر پنهان  $z$  یک متغیر تصادفی پیوسته<sup>۱۵</sup> است اما بدون از دست دادن عمومیت می توان مباحث مطرح شده را به متغیرهای گسسته نیز تعمیم داد [۵۱].

خروجی منعکس خواهد شد. بنابراین یک شبکه عصبی بازگشتی در نظر گرفته می شود که در هر لحظه از زمان خروجی سیستم را به عنوان ورودی می پذیرد و پارامترهای مدل را برمی گرداند. در [۳۷] و [۳۸] متغیرهای حالت و پارامترهای یک مدل فضای حالت غیرخطی با استفاده از روش های تغییراتی<sup>۱</sup> تخمین زده می شوند.

عملکرد مدل های یادگیری عمیق بر اساس گزارش های ارائه شده غیرقابل انکار است. اما استفاده از آنها در حوزه شناسایی سیستم ها نیز دارای معایبی هست. شبکه های عصبی عمیق به دلیل ماهیت فراپارامتری خود، عملکرد مناسبی دارد که این موضوع استفاده و استقرار آنها را برای کاربردهای بلادرنک محدود می کند [۳۹]. این محدودیت نه تنها در حوزه شناسایی سیستم ها بلکه در سایر حوزه ها نیز وجود دارد. بسته به محیط استقرار<sup>۲</sup> این مدل ها، روش های برای کاهش تعداد پارامترهای مدل و فشرده سازی<sup>۳</sup> مدل های آموزش دیده شده وجود دارد که امکان استفاده از این مدل ها را در سیستم هایی با منابع سخت افزاری محدود، مانند ظرفیت حافظه و توان پردازشی پردازنده در سیستم های یکپارچه<sup>۴</sup>، ممکن می سازد. این روش ها در حالت کلی به سه دسته ی کوانتیزه کردن<sup>۵</sup>، هرس کردن<sup>۶</sup> و تقطیر کردن دانش<sup>۷</sup> طبقه بندی می شوند [۴۰]، [۴۱]، [۴۲] و [۴۳]. مدل های یادگیری ماشین دارای دو جز ساختار و پارامتر هستند که پارامترهای مدل پس از آموزش به صورت اعشاری و در ۳۲ بیت<sup>۸</sup> ذخیره می شوند. کوانتیزه کردن دقت این پارامترها می توانند بعضاً تا اعداد صحیح و در ۸ بیت<sup>۹</sup> کاهش دهند که این کاهش دقت منجر به از دست رفتن عملکرد می شود. برای جلوگیری از این مسئله، کوانتیزه کردن مدل را حین آموزش انجام می دهند تا عملکرد مدل افت شدیدی نداشته باشد. روش های کوانتیزه کردن مدل ها در [۴۱]، [۴۴]، [۴۵]، [۴۶] و [۴۷] به تفصیل آورده شده است. دسته ای دیگر از روش های فشرده سازی مدل، ارتباطات اضافه و غیرمهم بین واحدهای محاسباتی گوناگون را می تواند حذف کند. این روش که موسوم به هرس کردن است با ایده از تکامل مغز انسان در مرحله نوزادی بیان می کند که تعداد ارتباطات عصبی بین نرون های مغز در یک نوزاد ابتدا پایین است که این تعداد در یک سن مشخص افزایش خواهد یافت و سپس دوباره کاهش می یابد. این کاهش ارتباطات عملکرد مغز را تحت تأثیر قرار نمی دهد [۴۸]. این روش ها در [۴۴] و [۴۷] به تفصیل ارائه شده است. تقطیر دانش که برای اولین بار در [۴۹] ارائه شده است بیان می کند که رفتار یک مدل عمیق با تعداد پارامترهای زیاد، توسط مدلهایی با تعداد پارامترهای

<sup>9</sup> Integer 8

<sup>10</sup> Benchmark

<sup>11</sup> Joint probability distribution

<sup>12</sup> Highly correlated high dimensional data

<sup>13</sup> Inference problem

<sup>14</sup> Latent variable models

<sup>15</sup> Continues random variable

<sup>1</sup> Variational methods

<sup>2</sup> Deployment environment

<sup>3</sup> Model compression

<sup>4</sup> Embedded systems

<sup>5</sup> Quantization

<sup>6</sup> Pruning

<sup>7</sup> Knowledge distillation

<sup>8</sup> Floating-point 32

$$\begin{aligned} E_{q\phi} \left( \log \frac{p_{\theta}(x, z)}{q_{\phi}(z|x)} \right) \\ = E_{q\phi} (\log p_{\theta}(x|z)) \\ - D_{kl} (q_{\phi}(z|x) \| p_{\theta}(z)) = F(\theta, \phi) \end{aligned} \quad (3)$$

که در آن  $D_{kl}$  معیار  $KL$ <sup>9</sup> است که برای محاسبه تفاوت بین دو توزیع احتمال به کار برده می شود. مسئله آموزش پارامترهای مسئله از طریق روش بیشینه سازی تابع شباهت که یک مسئله دشوار بود، تبدیل به یک مسئله بهینه سازی معادل با ترم های ساده تر شد. دلیل این سادگی حضور توزیع تغییراتی است که این امکان را به طراح می دهد تا به جای کارکرد با توزیع احتمال موخر<sup>11</sup> با یک توزیع احتمال ساده تر کار کند. حداکثر سازی عبارت موفق به منظور تعیین پارامترها معادل حداقل کردن  $D_{kl}$  است. لازم بذکر است که عبارت فوق زمانی حداقل خواهد شد که  $q_{\phi}(z|x)$  برابر با  $p_{\theta}(z|x)$  باشد.

$$\theta^*, \phi^* = \arg \max_{\theta, \phi} F(\theta, \phi) \quad (4)$$

چون دو مجموعه پارامترهای تحت بهینه سازی به یکدیگر وابسته هستند، برای بهینه سازی از روش بیشینه سازی امید تغییراتی<sup>12</sup> باید استفاده کنیم. بنابراین داریم:

$$\begin{aligned} E\text{-step:} \quad \phi_{k+1} &= \arg \max_{\phi} F(\theta_k, \phi) \\ M\text{-step:} \quad \theta_{k+1} &= \arg \max_{\theta} F(\theta, \phi_{k+1}) \end{aligned} \quad (5)$$

مسیر طی شده تا به اینجا کار یک مسیر استاندارد برای حل هر مسئله از طریق مدل متغیرهای پنهان است. تفاوت مسائل حل شده توسط این ابزار، در نحوه ی مدل کردن توزیع های شرطی موجود در روابط است. به عنوان مثال ماشین بولتزمن محدود شده<sup>13</sup> از مدل های مبتنی بر انرژی<sup>14</sup> برای مدل کردن  $p_{\theta}(x|z)$  و  $p_{\theta}(z|x)$  استفاده می کند که بدلیل فرضیات اعمال شده بر روی ساختار مدل، هر دو توزیع احتمال شرطی قابلیت ارزیابی و نمونه گیری را دارا هستند. دسته ی دیگری از رویکردهای مدل سازی از ظرفیت های شبکه های عصبی<sup>15</sup> در تقریب توابع استفاده می کند و توزیع های شرطی مورد نظر را تقریب می زند. در این رویکرد  $p_{\theta}(x|z)$  دارای یک فرم مشخص است که پارامترهای آن توسط یک شبکه عصبی با معماری مناسب تعیین می شود. چون این شبکه وظیفه باز تولید نمونه های  $x$  را دارد به عنوان شبکه کدگشا<sup>16</sup> گفته می شود. در این رویکرد چون

احتمال توام بر روی این متغیرها را می توان به صورت زیر تعریف کرد:

$$p(x, z) = p(z)p(x|z) \quad (1)$$

تعریف لایه پنهان به منظور کشف روابط همبستگی بین متغیرهای ورودی مسئله، این امکان را به ما می دهد که به جای رویارویی با توزیع پیچیده و نامشخص  $p(x)$  با یک توزیع احتمال مشخص و ساده، از نظر نمونه گیری<sup>1</sup> و ارزیابی، روبه رو شویم [19]. تمام اجزای این توزیع احتمال قابل تعیین و مشخص هستند. توزیع  $p(x)$  بر اساس این مدل به سادگی از طریق حاشیه سازی<sup>2</sup> بر روی متغیر  $z$  قابل تعریف است.

$$p(x) = \int p(x|z)p(z)dz \quad (2)$$

مسئله اصلی پیش روی این رویکرد این است انتگرال فوق قابل محاسبه<sup>3</sup> نیست و برای محاسبه پارامترهای مدل باید از روش های تقریب<sup>4</sup> استفاده کرد. در یک دسته بندی کلی روش های تقریب به دو گروه تقریب تصادفی<sup>5</sup> و تقریب معین<sup>6</sup> تقسیم بندی می شوند که در [13] و [19] به تفصیل آورده شده است. بدلیل بالا بودن بار محاسبات روش های تقریب تصادفی، در این مقاله از روش های تقریب معین تغییراتی استفاده شده است. لگاریتم تابع شباهت رابطه (2) به صورت زیر است:

$$\log p_{\theta}(x) = \log \int p_{\theta}(z)p_{\theta}(x|z)$$

که در آن  $\theta$  بردار مربوط به تمام پارامترهای متعلق به توزیع توام است. رابطه فوق را می توان به صورت زیر نوشت:

$$\log p_{\theta}(x) = \log \int \frac{p_{\theta}(x, z)}{q_{\phi}(z|x)} q_{\phi}(z|x) dz$$

که در آن  $q_{\phi}(z|x)$  یک توزیع شرطی معتبر است که در هیچ جای فضا و برای هیچ مقداری از  $z$  صفر نیست. این توزیع، توزیع تغییراتی<sup>7</sup> نامیده می شود. با استفاده از ناساوی<sup>8</sup> Jensen، یک کران پایین برای تابع بیشینه شباهت تعریف می شود [12]:

$$\begin{aligned} \log p_{\theta}(x) &\geq \int q_{\phi}(z|x) \log \frac{p_{\theta}(x, z)}{q_{\phi}(z|x)} dz \\ &= E_{q\phi} \left( \log \frac{p_{\theta}(x, z)}{q_{\phi}(z|x)} \right) \end{aligned}$$

رابطه فوق مشهور به کران پایین مشاهدات<sup>9</sup> است که به صورت زیر قابل بسط دادن است:

<sup>9</sup> Evidence lower bound (ELBO)

<sup>10</sup> Kullback-Leibler divergence

<sup>11</sup> Posterior probability distribution

<sup>12</sup> Variational EM

<sup>13</sup> Restricted Boltzmann machine (RBM)

<sup>14</sup> Energy-based models (EBM)

<sup>15</sup> Neural networks

<sup>16</sup> Decoder network

<sup>1</sup> Sampling

<sup>2</sup> Marginalization

<sup>3</sup> Intractable

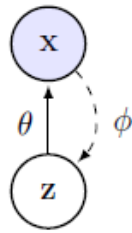
<sup>4</sup> Approximate method

<sup>5</sup> Stochastic approximate

<sup>6</sup> Deterministic approximate

<sup>7</sup> Variational distribution

<sup>8</sup> Jensen's inequality



شکل ۱- مدل مولد خودرمنگار تغییراتی [۱۷] - خطوط پررنگ مدل مولد خطوط منقطع مدل استنتاج یا شبکه بازشناسایی را نشان می دهد.

برای خودرمنگار تغییراتی توزیع کدکننده و کدگشا هر دو به صورت نرمال در نظر گرفته شده است که همین مسئله یکی از چالش های اساسی و محدود کننده استفاده از خودرمنگار تغییراتی است. زیرا بر اساس این فرض توزیع هایی که دارای مدهای بیشتری باشند قابلیت شناسایی ندارد و مهمتر اینکه کدکننده نمی تواند به صورت کامل توزیع احتمال موخر را تقریب بزند. با این حال خودرمنگار تغییراتی در طیف وسیعی از کاربردها با دقت خوبی به کار گرفته می شود. روابط مربوط به آموزش و تولید نمونه در این شبکه به صورت رابه (۶) خواهد بود:

$$\begin{aligned} q_{\phi}(z|x) &\sim N(\mu_{en}, \sigma_{en}^2) \\ &\rightarrow \begin{cases} \mu_{en} = NN_{en}^{\mu}(x) \\ \log \sigma_{en}^2 = NN_{en}^{\sigma}(x) \end{cases} \\ p_{\theta}(x|z) &\sim N(\mu_{de}, \sigma_{de}^2) \\ &\rightarrow \begin{cases} \mu_{de} = NN_{de}^{\mu}(z) \\ \log \sigma_{de}^2 = NN_{de}^{\sigma}(z) \end{cases} \end{aligned} \quad (6)$$

که در آن هر نمونه  $z$  به صورت  $z = \mu_{en} + \sigma_{en} \cdot \epsilon$  قابل محاسبه است که  $\epsilon$  یک توزیع نرمال استاندارد است. این رویکرد حقه دوباره پارامتریزه کردن<sup>۴</sup> نام دارد که چون نمونه گیری یک عمل گسسته است منجر به ایجاد مشکلات آموزشی با استفاده از گرادین نزولی خواهد شد که به این ترتیب دیگر مشکلی وجود نخواهد داشت. پارامترهای شبکه نیز با استفاده از حداقل کردن کران پایین مشاهدات بدست می آید که جزئیات مربوط به آن در [۱۶] آمده است. نکته مهم درباره آموزش مدل خود رمنگار تغییراتی این است که هر دو گام مربوط به بیشینه سازی امید تغییراتی به صورت همزمان انجام می شود و نه به صورت مجزا. این مسئله موجب می شود تا تخمین پارامترهای کدگشا به صورت زیربینه پیدا شود. زیرا آنچه که در بیشینه سازی تابع شباهت مشاهدات اثر دارد پارامترهای مربوط به کدگشاست و نه کدکننده. چون این دو مجموعه از پارامترها به صورت همزمان به روزرسانی می شوند، پارامترهای کدکننده نیز در بیشینه کردن تابع شباهت اثر خواهد داشت که نتیجه آن فاصله داشتن کران پایین مشاهدات با تابع شباهت مشاهدات است. برای رفع این مشکل، روشی به

توانایی مدل کردن توزیع احتمال موخر به صورت مستقیم وجود ندارد از یک شبکه بازشناسایی یا کدکننده استفاده می شود که وظیفه مدل کردن  $q_{\phi}(z|x)$  را برعهده دارد.

## ۲-۱- خودرمنگار تغییراتی

خودرمنگار تغییراتی یک مدل متغیر پنهان است که برای مدل کردن توزیع های احتمال شرطی بیان شده در بخش قبل از شبکه های عصبی استفاده می کند. قبل از بررسی نوع مدل کردن خود رمنگار تغییراتی لازم بذکر است که توزیع مولد برای یک مدل متغیر پنهان به صورت  $p(z)p(x|z)$  است که به منظور مشخص شدن کامل آن نیازمند هستیم تا هر کدام از این ترم ها دقیقاً تعریف شود. رابطه توزیع توام و ارتباط آن با توزیع حاشیه ای متغیرهای مسئله هیچ محدودیتی از جانب توزیع  $p(z)$  را ندارد. به بیان دیگر هر توزیع دلخواهی از  $p(z)$  می تواند در این مسئله جای داشته باشد اما بهترین نوع توزیع، توزیعی است که به ما اطمینان می دهد که مدل کدگشای  $p(x|z)$  می تواند بر اساس آن نمونه های مورد نظر ما را تولید کند. علاوه بر این مسئله، تعریف این ترم ها باید به نحوی باشد که فرآیند آموزش را تسهیل کند.

برای خودرمنگار تغییراتی توزیع لایه های مخفی به صورت یک توزیع نرمال مستقل از هم در نظر گرفته شده است. چرایی این مسئله ریشه در این حقیقت دارد که هر توزیع احتمال پیچیده ای را می توان بر اساس اعمال کردن یک نگاشت قطعی بر روی مجموعه ای از نمونه های توزیع استاندارد نرمال بدست آورد. به بیان بهتر اگر توانایی تولید نمونه از یک توزیع نرمال استاندارد وجود داشته باشد، هر توزیع دلخواهی را می توان بر اساس نگاشت این نمونه ها بدست آورد [۵۲]. بر اساس توضیحات ارائه شده انتظار می رود، فرآیند تولید نمونه توسط خودرمنگار تغییراتی از دقت خوبی برخوردار باشد زیرا توزیع لایه های کد یک توزیع نرمال است و نگاشت مطرح شده  $p(x|z)$  توسط یک شبکه عصبی مدل می شود که خاصیت تقریب عمومی<sup>۲</sup> را داراست.

تنها مسئله باقی مانده برای آموزش خودرمنگار تغییراتی وجود یک مکانیزم تولید نمونه برای متغیرهای  $z$  است که احتمال  $p(x|z)$  تحت نمونه های تولید شده مقداری بزرگتر از صفر داشته باشند. به بیان بهتر بر اساس داده های موجود مسئله باید فضای مجازی از یک توزیع نرمال استاندارد را انتخاب کرد که توزیع  $p(x|z)$  در آن فضا دارای احتمال بزرگتری است. برای این منظور یک شبکه بازشناسایی یا کدکننده در نظر گرفته شده است که بر اساس نمونه های ورودی  $x$  توزیعی از  $z$  را محاسبه می کند که بیشترین احتمال تولید  $x$  را داشته باشد. شکل ۱ مدل گرافی احتمالاتی<sup>۳</sup> خود رمنگار تغییراتی را نشان می دهد.

<sup>۳</sup> Probabilistic graphical model

<sup>۴</sup> Reparametrization trick

<sup>۱</sup> Recognition network – Encoder network

<sup>۲</sup> General approximation property

از دیدگاه یادگیری ماشین، آنالیز سری زمانی  $y_{1:T}$  نیازمند تعیین توزیع توام  $p(y_{1:T})$  بر روی تمام گام های زمانی<sup>۱۱</sup> است. در حالت کلی امکان مدل کردن این توزیع وجود ندارد زیرا همبستگی بین متغیرها بالا است و معمولاً برای مدل کردن آن باید از روابط استقلال شرطی<sup>۱۲</sup> و علیت در زمان<sup>۱۳</sup> استفاده کرد. زیرا اگر فرض کنیم هر متغیر  $x_t$  در هر لحظه از زمان یک متغیر دودویی<sup>۱۴</sup> باشد برای آنکه توزیع احتمال را بر روی کل دنباله محاسبه کنیم به تعداد  $2^T - 1$  پارامتر آزاد<sup>۱۵</sup> نیاز داریم تا توزیع احتمال به طور کامل مشخص شود که عملیاتی نیست. با استفاده از قانون احتمال شرطی بیز توزیع احتمال روی دنباله را می توان به صورت رابطه (۷) نوشت [۱۸]:

$$p(y_{1:T}) = \prod_{t=1}^T p(y_t | y_{1:t-1}) \quad (۷)$$

این نوع تجزیه کردن توزیع توام بر روی گام های زمانی منطبق بر علیت در زمان است زیرا هر فاکتور به تمام عوامل گذشته بستگی دارد. در هر صورت همچنان مدل کردن این دنباله کار دشواری است زیرا مجموعه متغیرهایی که در مجموعه ی شرط<sup>۱۶</sup> قرار گرفته اند با گذر زمان در حال تغییر است. به عنوان مثال اگر فرض کنیم می خواهیم این توزیع احتمال را با استفاده از یک شبکه عصبی مدل کنیم، نیازمند یک شبکه خواهیم بود که در هر لحظه از زمان ورودی های آن در حال تغییر است که این مسئله امری نشدنی است. بنابراین نیازمند ساده سازی و اعمال قیود بیشتری بر روی ساختار زمانی<sup>۱۷</sup> و ارتباط بین گام های زمانی متغیرها هستیم. این مسئله در واقع تفاوت بین ابزارهای متنوع که به منظور تحلیل سری های زمانی به کار می روند را نشان می دهد.

یکی از ساده سازی های صورت گرفته به منظور تحلیل محدود کردن وابستگی گام های زمانی<sup>۱۸</sup> به یکدیگر است. مثلاً به جای آنکه فرض کنیم هر گام زمانی به تمام گام های زمانی گذشته خود وابسته است، می توان فرض کرد که هر گام زمانی به یک طول مشخصی از گام های گذشته وابسته است. در این حالت توزیع توام به صورت رابطه (۸) خواهد بود:

$$p(y_{1:T}) = \prod_{t=1}^T p(y_t | y_{t-m:t-1}) \quad (۸)$$

این مدل را مدل مارکوف مرتبه  $m$ <sup>۱۹</sup> می نامند که چند نمونه از آن در شکل ۲ آورده شده است.

جهت محدود کردن کران پایین<sup>۱</sup> مشاهدات مطرح شد که با وزن دهی کردن به نمونه های تولید شده توسط کدکننده سعی در نزدیک کردن کران پایین مشاهدات و تابع شباهت نمونه ها دارد. مدل بدست آمده از این روش خودرمننگار وزن دهی شده اهمیت<sup>۲</sup> نام دارد که جزئیات مربوط به آن در [۵۳] آمده است. یکی دیگر از ایرادهای خودرمننگار تغییراتی، نوع مدل کردن کدکننده و کدگشا است که هر دو از یک توزیع نرمال پیروی می کنند. دلیل این نوع کردن تنها مسئله تسهیل آموزش را در بردارد. دو فاکتور اصلی و اساسی برای آموزش ساده تر شدن فرآیند اعمال گرادیان نزولی برای به روزرسانی وزن ها و نمونه گیری از توزیع کدگشا است. چون محاسبه  $D_{kl}$  بین دو توزیع نرمال دارای یک فرم مشخص و ساده است، استفاده از این فرض فرآیند آموزش را ساده می کند، بهمین دلیل چون طبق خاصیت نگاشت توزیع نرمال استاندارد امکان تغییر توزیع متغیرهای مخفی وجود ندارد و همچنین تغییر توزیع کدکننده منجر به دشواری کدگشا خواهد شد، چارچوب ارائه شده برای سایر توزیع های احتمال وجود ندارد. برای رفع این مشکل مدل خودرمننگار متخاصم<sup>۳</sup> ارائه شد که خودرمننگار تغییراتی را به عنوان یک حالت خاص در خود جای داده است [۵۴].

مباحث مطرح شده برای نوع داده های بدون حافظه<sup>۴</sup>، بدون هیچ تغییری قابل اعمال است و اصل ابزارها نیز برای این حالات ارائه شده است. به منظور توسعه این ابزار برای هدف شناسایی سیستم های غیرخطی، تعمیم ابزارهای موجود برای پردازش داده های دینامیکی امری ضروری خواهد بود.

## ۲-۲- مدل های متغیر پنهان ترتیبی<sup>۵</sup>

عمده اطلاعاتی که در محیط پیرامون ما وجود دارد پیوسته در حال تغییر است. این نوع اطلاعات به لحاظ ریاضی به صورت یک دنباله<sup>۶</sup> تعریف می شود. دنباله یک بردار معمولی با یک بعد فیزیکی<sup>۷</sup> است که این بعد فیزیکی در غالب سیستم ها به صورت زمان مدل می شود که به آن سری زمانی<sup>۸</sup> نیز گفته می شود [۱۸]. سری های زمانی در مقایسه با سیگنال های عادی دارای دو نوع اطلاعات هستند که یک نوع آن مربوط به مقدار عددی اطلاعات و دیگری اطلاعات زمانی<sup>۹</sup> سیگنال است. بنابراین روش های پردازش این نوع داده ها در مقایسه با ابزارهای پردازشی داده های ایستا پیچیده تر خواهد بود.

<sup>11</sup> Time steps

<sup>12</sup> Conditional independency

<sup>13</sup> Causal-in-time

<sup>14</sup> Binary variable

<sup>15</sup> Free parameters

<sup>16</sup> Conditioning set

<sup>17</sup> Temporal structure

<sup>18</sup> Temporal dependency

<sup>19</sup> mth-order Markov model

<sup>1</sup> ELBO lightening

<sup>2</sup> Importance weighted auto-encoders

<sup>3</sup> Adversarial auto-encoder

<sup>4</sup> Static data

<sup>5</sup> Sequential latent variable models

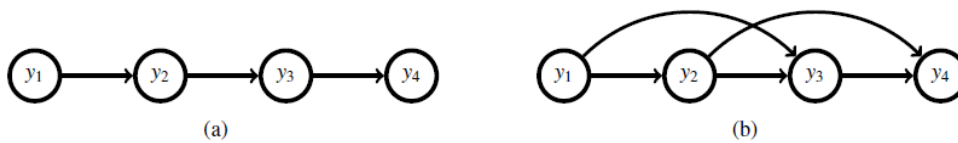
<sup>6</sup> Sequence

<sup>7</sup> Physical dimension

<sup>8</sup> Time series

<sup>9</sup> Temporal information

<sup>10</sup> Time series analysis



شکل ۲- شکل سمت چپ (a) - شبکه بیزین<sup>۱</sup> متناظر با مدل مارکوف مرتبه اول که در آن هر گام زمانی تنها به یک گام گذشته وابسته است. شکل سمت راست (b) - شبکه بیزین متناظر با مدل مارکوف مرتبه دو که در آن هر گام زمانی به دو گام زمانی قبل از خود وابسته است. [۱۸]

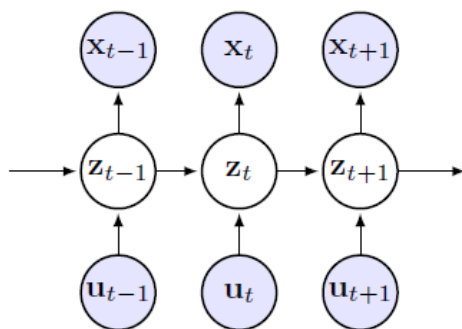
برای مدل کردن توزیع احتمال مورد نظر، مطابق آنچه که در مدل های متغیر پنهان بیان شد فرض می کنیم یک متغیر پنهان وجود دارد که وابستگی مستقیم به ورودی (یا گذشته خروجی در مدل های رگرسیون خود کار) دارد و همچنین به منظور حفظ اطلاعات زمانی به مقدار متغیر پنهان در گام قبل نیز وابسته است. در ادبیات مربوط به مدل های فضای حالت، به این متغیر پنهان، متغیر حالت<sup>۶</sup> گفته می شود. شکل ۳، مدل گرافی احتمالاتی این متغیر را نشان می دهد. بنابراین برای مدل کردن  $p_{\theta}(x_{1:T}|u_{1:T})$  خواهیم داشت [۱۸]:

$$p_{\theta}(x_{1:T}|u_{1:T}) = \int p_{\theta}(x_{1:T}, z_{1:T}|u_{1:T}) dz_{1:T} \quad (9)$$

که در آن فرض بر این است که متغیرهای حالت پیوسته هستند. در حالت گسسته انتگرال به جمع تبدیل می شود. توزیع احتمال توام  $p_{\theta}(x_{1:T}, z_{1:T}|u_{1:T})$  به صورت زیر قابل محاسبه است:

$$p_{\theta}(x_{1:T}, z_{1:T}|u_{1:T}) = p_{\theta}(z_1) \prod_{t=1}^T p_{\theta}(x_t|z_t) \prod_{t=2}^T p_{\theta}(z_t|z_{t-1}, u_t) \quad (10)$$

که در آن  $p_{\theta}(x_t|z_t)$  مدل مشاهدات<sup>۷</sup> و  $p_{\theta}(z_t|z_{t-1}, u_t)$  مدل انتقال حالت<sup>۸</sup> نامیده می شود. در برخی منابع مدل مشاهدات به ورودی نیز بستگی دارد ینی به صورت  $p_{\theta}(x_t|u_t, z_t)$  است.



شکل ۳- مدل فضای حالت [۱۷]

اگرچه این روش تا حد زیادی ساده سازی را امکان پذیر می کند، اما بزرگترین مشکل آن تعیین طول حافظه  $m$  است. به منظور افزایش قابلیت پیاده سازی معمولاً این مدل را برای کاربردهای که طول حافظه کوتاه باشد به کار می برند زیرا برای کاربردهایی که طول حافظه بزرگ نیاز دارند استفاده از این ابزار نیازمند تحلیل همبستگی<sup>۲</sup> و یا اطلاعات متقابل<sup>۳</sup> برای تعیین طول حافظه است. مدل های رگرسیون خود کار<sup>۴</sup> که در برخی از کتب به آن ها به عنوان یک ابزار تحلیل سری زمانی اشاره می شود، از زیر مجموعه های مدل های مارکوف است [۱۸].

یک چارچوب کلی تر و جامع تر برای مدل کردن سری های زمانی وجود دارد که در واقع ابزارهای موجود همگی به این فرم قابل تبدیل هستند [۱۸]. این چارچوب مدل های فضای حالت<sup>۵</sup> نام دارد. مدل های فضای حالت را می توان به نوعی انشقاق زمانی مدل های متغیر پنهان معرفی کرد. بر این اساس و طبق تعریف مدل های متغیر پنهان، می توان گفت مدل های فضای فرض می کند در هر گام زمانی یک مکانیزم پنهان وجود دارد که تغییرات موجود در متغیرها را مدل می کند اما این مکانیزم پنهان باید دارای یک وابستگی زمانی باشد تا بتوان اطلاعات موجود در متغیرها را نیز فرا بگیرد. در ادامه به بررسی این مدل ها می پردازیم.

### ۳-۲- مدل های فضای حالت

فرض می کنیم یک دنباله مشاهدات  $x_{1:T}$  وجود دارد که به یک مجموعه دنباله ورودی  $u_{1:T}$  وابسته است. در این حالت مسئله مدل کردن توزیع احتمال  $p_{\theta}(x_{1:T}|u_{1:T})$  است. لازم بذکر است که در بخش قبلی فرض بر این بود که مسئله یک مسئله رگرسیون خود کار است و تنها اطلاعاتی که در دست است مربوط به خود سری زمانی  $x_{1:T}$  است. فرم ذکر شده در بالا که سری زمانی مد نظر وابستگی به یک سری زمانی خارجی دارد فرم کلی تر مسئله مدل کردن سری زمانی است زیرا اگر ورودی مسئله را با گذشته متغیر  $x_t$  جایگزین کنیم، مدل رگرسیون خود کار حاصل می شود. بنابراین مباحث ذکر شده در ادامه، قابلیت تعمیم دادن به حالتی که ورودی محرک در مسئله وجود ندارد را هم داراست.

<sup>5</sup> State space models

<sup>6</sup> State variable

<sup>7</sup> Observation model (emission model)

<sup>8</sup> State transition model

<sup>1</sup> Bayesian network (Belief network)

<sup>2</sup> Correlation analysis

<sup>3</sup> Mutual information

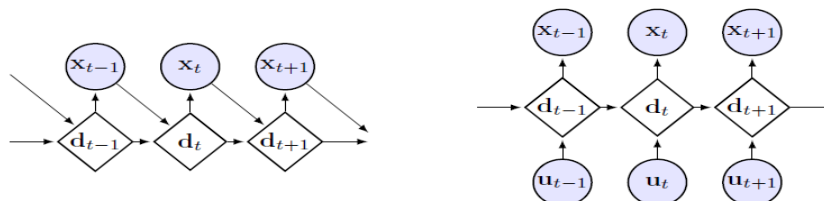
<sup>4</sup> Autoregressive models

فرض در دسترس بودن مدل فرض درستی نیست زیرا در طیف وسیعی از کاربردها مدل ارتباطی بین متغیرها در دست نیست.

جزئیات مربوط به مدل های خطی و مدل های غیرخطی که در آن استنتاج دقیق امکان پذیر است در [۵۶]، [۱۷] و [۱۸] آورده شده است. ظرفیت استفاده از این مدل ها در این نکته نهفته است که مدل های انتقال حالت و مدل مشاهدات توانایی تقریب زدن توسط هر تقریب زنی که باعث بهبود دقت در مسئله شود را داراست. در این راستا دسته ای از مدل ها در حال ظهور هستند که از ظرفیت شبکه های عصبی عمیق برای مدل کردن حالت و مشاهدات استفاده می کنند. به این دسته از مدل ها، مدل های فضای حالت عمیق<sup>۷</sup> گفته می شود. پر واضح است که در این مدل ها امکان استنتاج دقیق وجود ندارد و باید از روش های تقریبی برای استنتاج و آموزش آن استفاده کرد. در ادامه شبکه های عصبی بازگشتی<sup>۸</sup> به عنوان یک مدل مشهور فضای حالت مورد بررسی قرار می گیرد که از ظرفیت های مدل سازی آن در برخی از اقدامات این پژوهش استفاده شده است.

#### ۴-۲- شبکه های عصبی بازگشتی

شبکه های عصبی بازگشتی معماری خاصی از شبکه های عصبی هستند که توانایی مدل کردن دنباله با طول متغیر<sup>۹</sup> را توسط نگاشت های قطعی<sup>۱۰</sup> دارد [۵۵] و [۱۷]. شکل ۴ شبکه عصبی بازگشتی را برای دو حالت بدون ورودی و با ورودی نشان می دهد.



شکل ۴- شبکه عصبی بازگشتی - شکل سمت راست مربوط به حالتی است که ورودی محرک وجود دارد و شکل سمت راست مربوط به حالتی که ورودی محرک وجود ندارد. هر دو شماتیک یکسان هستند اگر در شماتیک سمت راست به جای ورودی محرک، خروجی لحظه قبل قرار بگیرد [۱۷].

معماری حافظه بلند کوتاه مدت<sup>۱۱</sup> و واحد بازگشتی دروازه ای<sup>۱۲</sup> از جمله این مدل ها هستند.

شبکه عصبی بازگشتی را می توان به صورت یک مدل فضای حالتی دید که توزیع احتمال انتقال حالت تابع دلتای دیراک است یعنی  $p_{\theta}(d_t | d_{t-1}, u_t) = \delta(d_t - f_{\theta}(d_{t-1}, u_t))$  مدل مشاهدات نیز در کاربردهای معمول به صورت یک نگاشت قطعی در نظر گرفته می شود، اما در برخی کاربردهای تولید متن مدل مشاهدات می تواند به صورت احتمالاتی در نظر گرفته شود [۶۰].

بسته به فرم های متفاوتی که برای مدل های انتقال حالت و مشاهدات در نظر گرفته می شود، مدل های فضای حالت متنوعی تولید خواهد شد که در ادامه به آن اشاره می شود. نکته مهم در این باره این است که هدف نهایی محاسبه توزیع شرطی دنباله خروجی به شرط ورودی است که این توزیع از طریق حاشیه سازی<sup>۱</sup> بر روی متغیرهای حالت بدست می آید. بنابراین نیازمند محاسبه  $p_{\theta}(z_{1:T} | x_{1:T}, u_{1:T})$  خواهیم بود. در حالت کلی محاسبه این توزیع، و یا بیان بهتر استنتاج دقیق<sup>۲</sup>، امکان پذیر نیست و تنها در شرایط خاص مسئله دارای جواب دقیق است [۱۷] و [۵۵]. به عنوان مثال اگر فرض کنیم مدل مشاهدات و مدل انتقال حالت در دسترس باشد و ارتباط بین متغیرها نیز خطی باشد، در شرایطی که هر دو توزیع احتمال انتقال حالت و مشاهدات گوسی باشد، مسئله استنتاج درباره متغیرهای حالت دارای یک فرم بسته خواهد بود که این فرم بسته به فیلتر کالمن<sup>۳</sup> مشهور است [۱۷]. در شرایطی که مدل ارتباطی بین متغیرهای حالت و متغیر خروجی به صورت یک تابع غیرخطی مشتق پذیر باشد، فیلتر کالمن به صورت تقریبی قابل استفاده را دارد که ابزار مورد نظر تحت عنوان های فیلتر کالمن خنثی<sup>۴</sup> و یا توسعه یافته<sup>۵</sup> مورد استفاده قرار می گیرند. برای طیف وسیعی از ابزارها مسئله دارای جواب دقیق نیست و باید از روش های تقریبی<sup>۶</sup> مانند روش های تغییراتی و یا نمونه گیری استفاده کرد. مهم تر اینکه

متغیرهای حالت در این مدل توسط یک نگاشت غیرخطی به صورت زیر محاسبه می شوند [۱۳]:

$$d_t = f_{\theta}(d_{t-1}, u_t; \theta) \quad (11)$$

انشقاق های متفاوتی از شبکه های عصبی بازگشتی وجود دارد که هر یک بسته به نوع کاربرد و در تقویت توانایی شبکه های عصبی بازگشتی به منظور مدل کردن حافظه توسعه پیدا کرده اند [۵۷]، [۵۸] و [۵۹]. تفاوت این ابزارها تنها در نحوه محاسبه متغیرها حالت است و در ماهیت یکسان هستند به این معنا که تمام آن ها یک نگاشت قطعی را مدل می کنند.

<sup>7</sup> Deep state space models

<sup>8</sup> Recurrent neural network

<sup>9</sup> Variable length sequence

<sup>10</sup> Deterministic maps

<sup>11</sup> Long short-term memory (LSTM)

<sup>12</sup> Gated recurrent unit (GRU)

<sup>1</sup> Marginalizing out

<sup>2</sup> Exact inference

<sup>3</sup> Kalman filter

<sup>4</sup> Unscented Kalman filter

<sup>5</sup> Extended Kalman filter

<sup>6</sup> Approximation method

## ۵-۲- استنتاج در مدل های فضای حالت غیرخطی

امکان استنتاج و آموزش مدل های زمانی در شرایط خاصی اعم از خطی بودن مدل ها و یا گوسی بودن توزیع های شرطی به صورت دقیق وجود دارد حال آنکه برای شرایطی که این فرضیات صادق نباشد، امکان یافتن پاسخ به صورت دقیق نیست و باید از روش های تقریبی استفاده کرد. بنابراین گام نخست، مدل کردن توزیع های مورد اشاره است. مطابق آنچه که در بخش مربوط به متغیرهای پنهان ایستا عنوان شد، مدل کردن توزیع های شرطی (فرض بر این است مدلی در دست نیست) می تواند بر مبنای مدل های مبتنی بر انرژی باشد و یا از ظرفیت شبکه های عصبی استفاده شود. این نوع مدل ها نیز از این قاعده مستثنی نیستند. برای این منظور می توان توزیع انتقال حالت و مشاهدات را به صورت یک فرم مشخص در نظر گرفت و سپس از ظرفیت شبکه های عصبی برای تخمین پارامتر استفاده کرد [۶۰]. روابط ریاضی مربوطه به صورت زیر خواهد بود:

$$\begin{aligned} p_{\theta}(z_t|z_{t-1}, u_t) &\sim N(\mu_{tr}, \Sigma_{tr}) \\ &\rightarrow \begin{cases} \mu_{tr} = NN_{tr}^{\mu}(z_{t-1}, u_t; \theta) \\ \log \Sigma_{tr} = NN_{tr}^{\sigma}(z_{t-1}, u_t; \theta) \end{cases} \\ p_{\theta}(x_t|z_t) &\sim N(\mu_{em}, \Sigma_{em}) \\ &\rightarrow \begin{cases} \mu_{em} = NN_{em}^{\mu}(z_t; \theta) \\ \log \Sigma_{em} = NN_{em}^{\sigma}(z_t; \theta) \end{cases} \end{aligned} \quad (12)$$

به منظور آموزش پارامترهای شبکه، تابع بیشینه شباهت را به صورت زیر می نویسم و با استفاده از روابط نوشته شده برای مدل متغیرهای پنهان ایستا در بخش قبلی، کران پایین تابع شباهت را محاسبه می کنیم [۵۵]:

$$\begin{aligned} \log p_{\theta}(x_{1:T}|u_{1:T}) &\geq \int q_{\phi}(z_{1:T}|x_{1:T}, u_{1:T}) \log \frac{p_{\theta}(x_{1:T}, z_{1:T}|u_{1:T})}{q_{\phi}(z_{1:T}|x_{1:T}, u_{1:T})} dz_{1:T} \\ &= E_{q_{\phi}} \left( \log \frac{p_{\theta}(x_{1:T}, z_{1:T}|u_{1:T})}{q_{\phi}(z_{1:T}|x_{1:T}, u_{1:T})} \right) \end{aligned}$$

گام بعدی تعیین ساختار توزیع تغییراتی است که هیچ محدودیتی برای آن وجود ندارد اما دقت روش های تقریبی به شدت به نوع انتخاب این توزیع بستگی دارد. تنها نکته ای که باید برای تعریف آن لحاظ کرد مسئله تسهیل فرآیند آموزش مدل است. بهترین انتخاب برای توزیع تغییراتی در روش های بهینه سازی تغییراتی استفاده از ساختار خود مسئله و تحمیل کردن ساختار مسئله به این توزیع است. به بیان بهتر برای آنکه بهترین انتخاب را برای توزیع تغییراتی داشته باشیم لازم است  $p_{\theta}(z_{1:T}|x_{1:T}, u_{1:T})$  را تعیین کنیم زیرا توزیع تغییراتی مورد نظر، تقریبی از این توزیع احتمال موخر را بدست می دهد (فرآیند کمینه سازی کران پایین مشاهدات در جایی حداقل است که توزیع تغییراتی برابر با توزیع احتمال موخر باشد). بنابراین خواهیم داشت:

$$\begin{aligned} p_{\theta}(z_{1:T}|x_{1:T}, u_{1:T}, z_0) \\ = p_{\theta}(z_{2:T}|z_1, x_{1:T}, u_{1:T}, z_0) p_{\theta}(z_1|x_{1:T}, u_{1:T}, z_0) \end{aligned}$$

که در آن  $p_{\theta}(z_{2:T}|z_1, x_{1:T}, u_{1:T})$  را بر اساس قاعده استقلال شرطی می توان به صورت  $p_{\theta}(z_{2:T}|z_1, x_{2:T}, u_{2:T})$  نوشت. با ادامه دادن این روند برای تمام گام های زمانی خواهیم داشت:

$$p_{\theta}(z_{1:T}|x_{1:T}, u_{1:T}, z_0) = \prod_{t=1}^T p_{\theta}(z_t|z_{t-1}, x_{t:T}, u_{t:T})$$

بنابراین ساختار توزیع تغییراتی نیز مطابق با همین الگو به صورت زیر قابل تجزیه خواهد بود:

$$\begin{aligned} q_{\phi}(z_{1:T}|x_{1:T}, u_{1:T}, z_0) \\ = \prod_{t=1}^T q_{\phi}(z_t|z_{t-1}, x_{t:T}, u_{t:T}) \end{aligned} \quad (13)$$

کران بالای مشاهدات نیز به صورت زیر خواهد بود:

$$\begin{aligned} \log p_{\theta}(x_{1:T}|u_{1:T}) &= E_{q_{\phi}} \left( \log \frac{p_{\theta}(x_{1:T}, z_{1:T}|u_{1:T})}{q_{\phi}(z_{1:T}|x_{1:T}, u_{1:T})} \right) \\ &= \sum_{t=1}^T E_{q_{\phi}^*(z_{t-1})} \left( E_{q_{\phi}(z_t|z_{t-1}, x_{t:T}, u_{t:T})} (\log p_{\theta}(x_t|z_t)) \right. \\ &\quad \left. - D_{kl}(q_{\phi}(z_t|z_{t-1}, x_{t:T}, u_{t:T}) || p_{\theta}(z_t|z_{t-1}, u_t)) \right) \end{aligned}$$

که در آن  $q_{\phi}^*(z_{t-1})$  توزیع حاشیه ای توزیع تغییراتی حول متغیر  $z_{t-1}$  است. واضحاً رابطه بدست آمده مشابه با رابطه بدست آمده برای خودرگرار تغییراتی است با این تفاوت که توزیع پیشین<sup>۱</sup> با زمان در حال تغییر کردن است.

پس از مشخص شدن تابع هزینه، نوبت به مدل کردن شبکه بازشناسایی یا کدکننده می رسد. همانطور که مشخص است، مدل کردن شبکه کدکننده به سادگی امکان پذیر نیست زیرا به مقادیر آینده ورودی و خروجی بستگی دارد. برای رفع این مشکل دو روش در حالت کلی وجود دارد. در روش اول می توان فرض کرد که کل دنباله در اختیار نیست و داده ها با گذر زمان در اختیار مدل قرار می گیرند. این رویکرد فیلترینگ نام دارد و چون در آن از شبکه عصبی استفاده می شود به آن فیلتر کالمن عمیق<sup>۲</sup> گفته می شود [۱۷]، [۵۵] و [۶۱]. بر اساس این رویکرد یک متغیر کمکی  $a_t$  تعریف می شود که وابستگی مستقیم به ورودی و خروجی در زمان یکسان دارد و متغیر حالت تابعی از این متغیر کمکی و متغیر پیشین حالت خواهد بود [۶۱]. به بیان ریاضی خواهیم داشت:

$$\begin{aligned} q_{\phi}(z_{1:T}|x_{1:T}, u_{1:T}, z_0) \\ = \prod_{t=1}^T q_{\phi}(z_t|z_{t-1}, a_t), \\ a_t = f_{\lambda}(x_t, u_t) \end{aligned} \quad (14)$$

مدل گرافی متناظر با این مدل به صورت شکل ۵ خواهد بود.

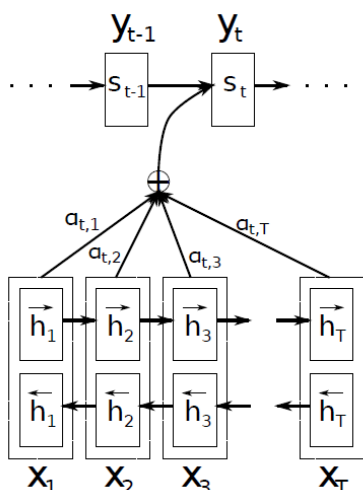
<sup>2</sup> Deep Kalman Filter (DKF)

<sup>1</sup> Prior distribution

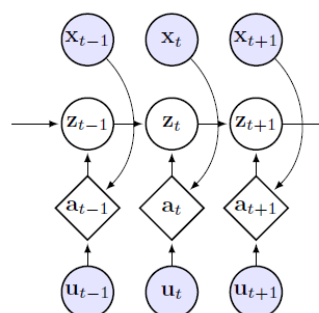
همانطور که مشاهده شد، مدل فضایی حالت غیرخطی دارای یک فرآیند آموزش قابل اجرا است اما دارای ایراداتی نیز هست. اولین ایراد مربوط به بار پردازشی بالا این مدل است که در آن یک شبکه عصبی بازگشتی مورد استفاده قرار می گیرد و در شرایطی که مدل نیازمند عمق بالایی دینامیک<sup>۳</sup> باشد، مدت زمان زیادی طول می کشد تا کل دنباله ورودی و خروجی پردازش شود. این بار پردازشی در شرایطی که در شبکه بازشناسایی از یک شبکه عصبی معکوس استفاده کنیم دو چندان خواهد شد. ایراد مهم بعدی ناکارآمد بودن این مدل در شرایطی است که ابعاد مسئله بالا باشد. برای حل مشکل اول و دوم که عدم توانایی پردازش دنباله های با عمق بالا و نیازمند تحلیل دستی برای تعیین عمق است می توان از ترنسفرمرها<sup>۴</sup> و مکانیزم توجه<sup>۵</sup> استفاده کرد که قادر است در یک فرآیند انتها به انتها و با استفاده از مکانیزم توجه متغیرهای حالت را به صورت یکجا محاسبه کند. بنابراین این امکان وجود دارد که در این مدل ها، به جای استفاده از شبکه های عصبی عمیق از ترنسفرمرها استفاده کرد.

## ۶-۲- مکانیزم توجه<sup>۶</sup>

مدل هایی که تاکنون بررسی شد، برای فراگرفتن اطلاعات زمانی از شبکه های عصبی بازگشتی استفاده می کنند. بزرگترین ایراد این شبکه ها در ساختار کدکننده-کدگشا است که لایه میانی این مدل ها وظیفه خلاصه کردن اطلاعات مربوط به دنباله ها را به تنهایی برعهده دارد. در شرایطی که طول دنباله ها بزرگ باشد، این لایه توانایی خلاصه سازی به صورت صحیح را نخواهد داشت [۶۲].

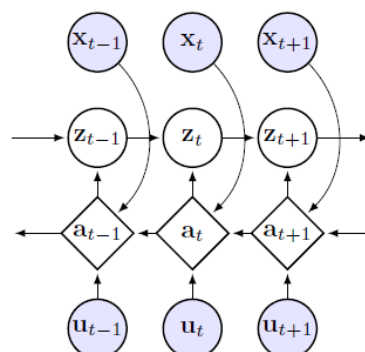


شکل ۷- ساختار کدکننده - کدگشا در حضور مکانیزم توجه [۶۲]



شکل ۵ کالمن فیلتر عمیق به منظور رفع مشکل وابستگی متغیرهای حالت به مقادیر آینده ورودی و خروجی در شبکه بازشناسایی توسعه داده شده [۱۷]

رویکرد غالب دیگری نیز وجود دارد که از یک شبکه عصبی بازگشتی معکوس<sup>۱</sup> استفاده می کند. بر اساس این رویکرد فرض بر این است که دنباله ورودی و خروجی به طور کامل در اختیار است و شبکه با مقدار نهایی این دنباله ها کار خود را آغاز می کند و خلاصه ای از مقادیر آینده را در اختیار شبکه بازشناسایی قرار می دهد. اگرچه دقت این رویکرد از رویکرد فیلتر کردن بالاتر است، بار محاسباتی آن نیز بالاتر است زیرا امکان پردازش موازی وجود ندارد و تا به طور کامل کل دنباله در جهت عکس پردازش نشود امکان تولید نمونه های بعدی برای متغیرهای حالت وجود ندارد. این رویکرد هموارساز کالمن عمیق<sup>۲</sup> نامیده می شود که در شکل ۶ به نمایش در آمده است.



شکل ۶- هموارساز کالمن عمیق به منظور خلاصه سازی گام های زمانی

آینده برای شبکه بازشناسایی [۱۷]

روابط مربوط به این شبکه نیز به صورت زیر خواهد بود [۶۱]:

$$q_{\phi}(z_{1:T}|x_{1:T}, u_{1:T}, z_0) = \prod_{t=1}^T q_{\phi}(z_t|z_{t-1}, a_t), \quad (15)$$

$$a_t = f_{\lambda}(a_{t+1}, [x_t, u_t])$$

نتیجتاً شبکه بازشناسایی به صورت زیر مدل خواهد شد:

$$q_{\phi}(z_t|z_{t-1}, a_t) \sim N(\mu_{en}, \Sigma_{en})$$

$$\rightarrow \begin{cases} \mu_{en} = NN_{en}^{\mu}(z_{t-1}, a_t; \phi) \\ \log \Sigma_{en} = NN_{en}^{\sigma}(z_{t-1}, a_t; \phi) \end{cases} \quad (16)$$

<sup>4</sup> Transformers

<sup>5</sup> Attention mechanism

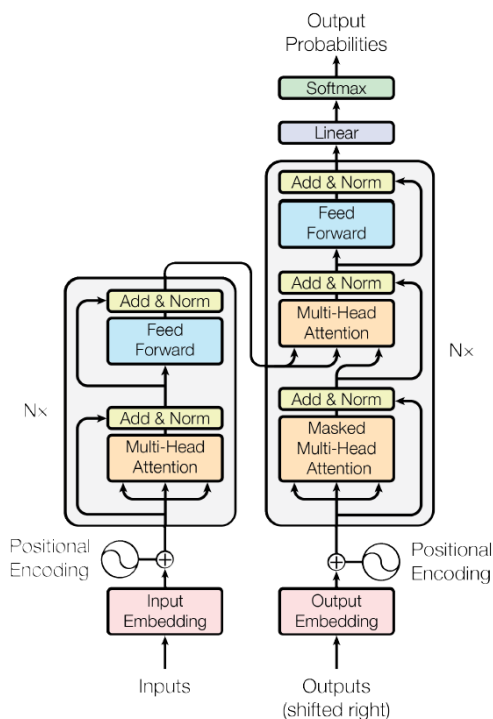
<sup>6</sup> Attention mechanism

<sup>1</sup> Reversed RNN

<sup>2</sup> Deep Kalman Smoother (DKS)

<sup>3</sup> Large dynamic range

ترنسفرمرها جایجا شد که موجبات مدل سازی های گسترده در حوزه پردازش زبان طبیعی<sup>۴</sup> را فراهم کرد. بخش اصلی ترنسفرمرها که در شکل ۸ به نمایش درآمده است که مکانیزم خود-توجه است. بر اساس این مکانیزم، بازنمایی که برای هر کلمه در وظیفه ماشین ترجمه استخراج می-شود، حساسیتی برای ترتیب کلمات حاضر در یک دنباله متنی ورودی قائل نیست. به بیان بهتر، بازنمایی هر کلمه تنها به حضور کلمات حساس است. لازم بذکر است امروزه استفاده از ترنسفرمرها صرفا معطوف به حوزه پردازش زبان طبیعی و مدل سازی داده های ترتیبی نیست و در حوزه بینایی ماشین نیز از این ابزار استفاده می شود [۶۴]، [۶۵]، [۶۶] و [۶۷].



شکل ۸- ساختار کدکننده-کدگشا در حضور مکانیزم خودتوجه - ترنسفرمر [۶۳]

بر اساس این مدل، ابتدا بازنمایی هایی که از هر کلمه استخراج می-شود، از یک مکانیزم توجه عبور می کند که این مکانیزم یک رابطه ساده محاسبه ضرب داخلی مقیاس شده<sup>۵</sup> است. ریشه ی اصلی این مکانیزم مربوط به سیستم های بازیابی اطلاعات است [۶۳]. در یک سیستم بازیابی اطلاعات<sup>۶</sup> زمانی که کاربر یک عبارت جستجو<sup>۷</sup> را وارد می کند، سیستم بر اساس اطلاعاتی که در پایگاه داده خود دارد به هر مقدار<sup>۸</sup> یک کلید<sup>۹</sup> اختصاص می دهد و بر اساس شباهتی<sup>۱۰</sup> که بین عبارت جستجو و کلید وجود دارد، مقدار متناظر را برمی گرداند. بر این اساس، کارکرد مدل بر

برای مدل کردن این لایه از یک شبکه عصبی بازگشتی استفاده می-شود که گام آخر مربوط به متغیر حالت شبکه به عنوان بردار زمینه<sup>۱</sup> در نظر گرفته می شود. به منظور رفع این مشکل در [۲۹] یک فرآیندی تحت عنوان مکانیزم-توجه معرفی شده است که بردار زمینه را بر اساس میانگین وزن دار متغیرهای حالت شبکه عصبی در تمام گام های زمانی محاسبه می کند. شکل ۷ ساختار مربوط به این مدل را نشان می دهد. بر اساس این ساختار، کدکننده ابتدا اطلاعات زمانی مربوط به دنباله ورودی را استخراج می کند و برای هر گام از اطلاعات زمانی استخراج شده توسط کدکننده یک مقدار اهمیت برای هر متغیر حالت شبکه عصبی بازگشتی کدکننده محاسبه می کند تا گام بعدی متغیر حالت را برای شبکه عصبی بازگشتی به کار گرفته شده برای کدگشا محاسبه کند [۶۲]. این مقدار اهمیت توسط یک شبکه عصبی مدل می شود که در ادبیات یادگیری ماشین به آن توجه جمع شونده<sup>۲</sup> نیز گفته می شود. مقادیر اهمیت مطابق با رابطه (۱۷) ساخته می شوند.

$$e_{ij} = NN(s_{i-1}, h_j), \alpha_{ij} = \frac{\exp(e_{ij})}{\sum_j \exp(e_{ij})} \quad (17)$$

که در آن  $e_{ij}$  شباهت بین حالت آم کدگشا با حالت آم کدکننده را محاسبه می کند. بردار زمینه در این حالت به صورت رابطه (۱۸) مدل می شود.

$$c_i = \sum_j \alpha_{ij} h_j \quad (18)$$

پس از محاسبه این بردار، حالت بعدی کدگشا با استفاده از یک شبکه عصبی بازگشتی و در شرایطی که بردار زمینه و مقدار واقعی خروجی در گام قبلی، به عنوان ورودی شبکه لحاظ می شوند. این مسئله مشکل عدم توانایی شبکه های عصبی بازگشتی برای خلاصه کردن دنباله ورودی در یک متغیر با بعد محدود را مرتفع می کنند. اما دارای یک چالش بزرگ است که مربوط به عدم توانایی شبکه های عصبی بازگشتی در مدل کردن دنباله به صورت موازی هستند. شبکه های عصبی بازگشتی، دنباله ورودی را به صورت ترتیبی پردازش می کنند. برای حل این مشکل [۶۳] یک مکانیزمی به نام خود-توجه<sup>۳</sup> را معرفی می کند که زمینه حذف کردن لایه-های بازگشتی در شبکه های عصبی را فراهم می کند.

## ۸-۲- مکانیزم خود-توجه و ترنسفرمرها

ساختار کدکننده-کدگشا که پیش تر دارای از دو شبکه عصبی بازگشتی تشکیل می شد، توسط مولفین [۶۳] با شبکه ای تحت عنوان

<sup>6</sup> Information retrieval system

<sup>7</sup> Query

<sup>8</sup> Value

<sup>9</sup> Key

<sup>10</sup> Similarity

<sup>1</sup> Context vector

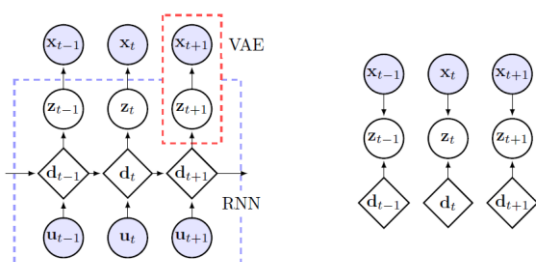
<sup>2</sup> Additive attention

<sup>3</sup> Self-attention

<sup>4</sup> Natural language processing (NLP)

<sup>5</sup> Scaled dot-product

کردن سیگنال گفتار نیازمند دو نوع نگاهت غیرخطی است که نگاهت اول یک نگاهت معین و نگاهت دوم یک نگاهت تصادفی است. با همین انگیزه می توان مسئله بیان شده را به حوزه شناسایی سیستم های غیرخطی نیز تعمیم داد چرا که هر دنباله ورودی و خروجی دارای اشتراکاتی هستند که مربوط به رابطه بین ورودی و خروجی است و از طرفی دارای تغییراتی نیز هستند که برای هر دنباله ورودی و خروجی منحصر بفرد است (با این فرض که این تغییرات به صورت ایستا مدل می شود و نه غیر ایستا که خود این مطلب یکی از ایرادات این معماری است زیرا فرض می کند که متغیرهای تصادفی هیچ ارتباطی با هم ندارند). این مسئله برای شناسایی سیستم های غیرخطی در [۳۳] بررسی شده است. شکل ۹ مدل گرافی احتمالاتی این شبکه را برای هر دو شبکه مولد و شبکه بازشناسایی به تصویر کشیده است.



شکل ۹- مدل گرافی احتمالاتی مربوط به ساختار VAE-RNN [۱۷]

در این معماری ابتدا به کمک یک شبکه عصبی بازگشتی، یک بازنمایی از داده های ورودی استخراج می شود که این بازنمایی مربوط به اطلاعات زمانی متغیرهای ورودی است. سپس بازنمایی استخراج شده به عنوان ورودی یک خود رمزنگار تغییراتی عمل می کند و بخش کدگشای مدل سعی می کند اطلاعات مربوط به خروجی را از روی اطلاعات کد شده ورودی، تولید کند. مزیت این رویکرد استخراج هر دو مجموعه ویژگی مرتبط با تغییرات متغیرهای ورودی و ویژگی های قطعی منحصر به فرد سیگنال ورودی است که توانایی مدل کردن توزیع شرطی  $p_{\theta}(x_{1:T}|u_{1:T})$  داراست. توزیع توام برای مدل مولد به صورت زیر

خواهد بود:

$$p_{\theta}(x_{1:T}, d_{1:T}, z_{1:T}|u_{1:T}) = \prod_{t=1}^T p_{\theta}(z_t|d_t) p_{\theta}(d_t|d_{t-1}, u_t) p_{\theta}(x_t|z_t) \quad (19)$$

که در آن  $\delta(d_t - \tilde{d}_t)$ ,  $\tilde{d}_t = f_{\theta}(d_{t-1}, u_t)$  است که با توجه به خاصیت غربالی تابع ضربه خواهیم داشت:

محاسبه سه ویژگی جستجو، مقدار و کلید استوار است که به نسبت شباهتی که بین ویژگی های جستجو و کلید وجود دارد، یک نگاهت مناسب با مقدار را برمی گرداند. ویژگی های مقدار، کلید و جستجو هر سه از یک تبدیل خطی بدست می آیند و سپس مطابق با رابطه (۱)، مقدار اهمیت هر کلمه مشخص می شود.

$$o = \text{softmax}\left(\frac{Q^T K}{\sqrt{d}}\right) v \quad (18)$$

نکته مهم درباره ترنسفررها این است که هیچ لایه بازگشتی وجود ندارد و مدل از طریق بردار کدکننده موقعیت<sup>۱</sup>، اطلاعات زمانی را فرا می گیرد. در [۶۳] از توابع مثلثاتی برای کد کردن موقعیت استفاده شده است. رویکردهای متفاوتی برای محاسبه کدکننده موقعیت وجود دارد که در [۶۸] آورده شده است. روند مشابه با روند معرفی شده برای کدکننده در کدگشا اتفاق می افتد با این تفاوت که چون قصد داریم مقدار آینده خروجی را پیش بینی کنیم، از مکانیزم خود-توجه پوشش داده شده برای کدگشا استفاده می شود تا مقادیر آینده خروجی در مدل سازی لحاظ نگردد. جزئیات مربوط به ترنسفررها در [۶۳] آمده است.

### ۳- مروری بر ساختارهای مدل های دینامیکی

#### عمیق در شناسایی سیستم ها

هیچ یک ابزارهایی که از حوزه یادگیری ماشین برای توسعه روش های شناسایی غیرخطی به کار گرفته شده است، توسط جوامع مهندسی کنترل توسعه نیافته است و عمده این روش ها برای منظور شناسایی گفتار و پردازش متن توسعه یافته است [۱۷]. برخی از این پژوهش ها از خود رمزنگار تغییراتی و یا ترکیبی از خود رمزنگار تغییراتی و شبکه های عصبی بازگشتی به منظور یک ابزار برای شناسایی فضای حالت غیرخطی و یا خطی استفاده کرده اند [۳۲]، [۳۳]، [۳۶]، [۳۴]، [۶۹]، [۷۰] و [۷۱]. چون در بخش قبل، خود رمزنگار تغییراتی معرفی شد، در این بخش ابزارهایی که از ترکیب خود رمزنگار تغییراتی و شبکه های عصبی بازگشتی به وجود آمده اند، معرفی خواهد شد.

#### ۳-۱- خود رمزنگار تغییراتی - شبکه عصبی بازگشتی<sup>۲</sup>

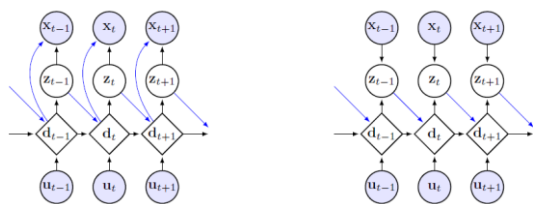
خود رمزنگار تغییراتی - شبکه عصبی بازگشتی که در [۱۷] معرفی شده است با هدف اولیه تولید متن و گفتار<sup>۳</sup> توسعه داده شده است که قادر است از دو منظر یک مدل مولد گفتار<sup>۴</sup> تولید کند. بر اساس این مدل، گفتار هر شخص دارای دو منبع اطلاعات است. منبع اول مربوط به مشخصات زبانی گوینده است که بین تمام گوینده های آن زبان یکسان است و منبع دوم مربوط به تغییراتی است که از هر فرد به فرد دیگر متفاوت است. از دیدگاه ریاضی می توان این مسئله را به این صورت عنوان کرد که مدل

<sup>3</sup> Speech

<sup>4</sup> Speech generative model

<sup>1</sup> Positional encoding

<sup>2</sup> VAE-RNN



شکل ۱۰ - شبکه های مولد و بازشناسایی مدل عصبی بازگشتی تغییراتی

[۱۷]

مطابق با این ساختار، ویژگی های قطعی محاسبه شده توسط شکل ۲-۸ تابعی از ویژگی های تصادفی نیز بوده و خروجی نیز بر اساس ویژگی های تصادفی و قطعی ساخته می شود. از این ساختار در [۱۸] به منظور شناسایی سیستم های غیرخطی استفاده شده است.

روابط مربوط به شبکه مولد به صورت زیر خواهد بود:

$$p_{\theta}(x_{1:T}, z_{1:T}, d_{1:T} | u_{1:T}) = \prod_{t=1}^T p_{\theta}(x_t | z_t, d_t) p_{\theta}(d_t | z_{t-1}, d_{t-1}, u_t) p_{\theta}(z_t | d_t) \quad (24)$$

که در آن  $\tilde{d}_t = d_t - \tilde{d}_{t-1}$ ،  $p_{\theta}(d_t | d_{t-1}, z_{t-1}, u_t) = \delta(d_t - \tilde{d}_t)$  چون متغیرهای حالت شبکه عصبی بازگشتی به  $z_{t-1}$  وابسته است دیگر ماهیت قطعی بودن خود را از دست می دهد و دارای عدم قطعیت نیست. اما این مطلب درست نیست زیرا وابستگی به  $z_{t-1}$  ای که به لحاظ زمانی رخ داده است به صورت قطعی است که از گام پیشین زمانی نمونه گیری شده است. با توجه به خاصیت غربالی تابع ضربه خواهیم داشت:

$$p_{\theta}(x_{1:T}, z_{1:T}, d_{1:T} | u_{1:T}) = \prod_{t=1}^T p_{\theta}(x_t | z_t, \tilde{d}_t) p_{\theta}(z_t | \tilde{d}_t) \quad (25)$$

به منظور آموزش پارامترهای مدل مشابه حالت قبل، کران پایین مشاهدات را به صورت زیر می نویسیم:

$$\log p_{\theta}(x_{1:T} | u_{1:T}, d_0) \geq E_q \left( \log \frac{p_{\theta}(x_{1:T}, z_{1:T} | u_{1:T}, d_0)}{q_{\phi}(z_{1:T} | x_{1:T}, u_{1:T}, d_0)} \right) = F(\theta, \phi)$$

که در آن توزیع تغییراتی، مطابق آنچه که در بخش مربوط به مدل متغیرهای پنهان گفته شود بر اساس ساختار واقعی توزیع احتمال موخر به صورت زیر قابل تجزیه است:

$$q_{\phi}(z_{1:T} | x_{1:T}, u_{1:T}) = \prod_{t=1}^T q_{\phi}(z_t | x_t, \tilde{d}_t) \quad (26)$$

با این تعریف کران پایین مشاهدات به صورت زیر خواهد بود:

$$p_{\theta}(x_{1:T}, z_{1:T} | u_{1:T}) = \prod_{t=1}^T p_{\theta}(z_t | \tilde{d}_t) p_{\theta}(x_t | z_t) \quad (20)$$

به منظور آموزش پارامترهای مدل مشابه حالتی که برای فضای حالت

غیرخطی دیدیم، کران پایین مشاهدات را به صورت زیر می نویسیم:

$$\log p_{\theta}(x_{1:T} | u_{1:T}, d_0) \geq E_q \left( \log \frac{p_{\theta}(x_{1:T}, z_{1:T} | u_{1:T}, d_0)}{q_{\phi}(z_{1:T} | x_{1:T}, u_{1:T}, d_0)} \right) = F(\theta, \phi) \quad (21)$$

که در آن توزیع تغییراتی، بر اساس ساختار واقعی توزیع احتمال موخر

به صورت رابطه (۲۲) قابل تجزیه است:

$$q_{\phi}(z_{1:T} | x_{1:T}, u_{1:T}) = \prod_{t=1}^T q_{\phi}(z_t | x_t, \tilde{d}_t) \quad (22)$$

با این تعریف کران پایین مشاهدات به صورت زیر خواهد بود:

$$F(\theta, \phi) = \sum_{t=1}^T E_{q_{\phi}} [\log p_{\theta}(x_t | z_t)] - D_{kl} (q_{\phi}(z_t | x_t, \tilde{d}_t) \| p_{\theta}(z_t | \tilde{d}_t)) \quad (23)$$

تابع هزینه مناسب برای آموزش این مدل، مشخصاً شباهت نزدیک تابع هزینه و مکانیزم آموزش خود رمزنگار تغییراتی دارد که نتیجتاً به سادگی و با استفاده از الگوریتم گرادینان نزولی قابل بهینه سازی است. تمامی ترم های توزیع شرطی موجود در این مدل از یک توزیع گوسی پیروی می کند که پارامترهای آن توسط یک شبکه عصبی تعیین می شود. بزرگترین ایراد مدل توسعه داده شده در این است که خروجی مدل تنها تابعی از بخش تصادفی ویژگی های ورودی است. در حالی که هر سیگنال زمانی را می توان به دو مولفه تصادفی و قطعی تجزیه کرد. بنابراین لازم است که یک ارتباط مستقیم از ویژگی های قطعی به خروجی مدل وجود داشته باشد. ساختار پیدا شده از این طریق شبکه عصبی بازگشتی تغییراتی<sup>۱</sup> نامیده می شود که در ادامه مورد بررسی قرار می گیرد.

## ۲-۳- شبکه عصبی بازگشتی تغییراتی

یکی از ایرادات ساختار خودرمزنگار تغییراتی- شبکه عصبی بازگشتی تولید خروجی تنها بر اساس ویژگی های تصادفی است حال آنکه خروجی از ویژگی های قطعی نیز تاثیر می پذیرد [۱۷] و [۵۵]. بنابراین لازم است تا هر دو مورد ذکر شده در ساختار بررسی شده در بخش ۳-۱ نیز لحاظ شود. تغییرات اعمال شده بر روی ساختار خودرمزنگار تغییراتی - شبکه عصبی بازگشتی منجر به تولد یک مدل فضای حالت به نام شبکه عصبی بازگشتی تغییراتی می شود [۷۲] که شبکه های مولد و بازشناسایی آن مطابق شکل ۱۰ خواهد بود.

<sup>1</sup> Variational RNN

$$p_{\theta}(x_{1:T}, z_{1:T}, d_{1:T} | u_{1:T}) \quad (28)$$

$$= \prod_{t=1}^T p_{\theta}(x_t | z_t, d_t) p_{\theta}(d_t | d_{t-1}, u_t) p_{\theta}(z_t | d_t, z_{t-1})$$

که در آن  $\tilde{d}_t = d_t - \bar{d}_t$ ،  $p_{\theta}(d_t | d_{t-1}, u_t) = \delta(d_t - \bar{d}_t)$  است که با توجه به خاصیت غربالی تابع ضربه خواهیم داشت:

$$p_{\theta}(x_{1:T}, z_{1:T} | u_{1:T}) \quad (29)$$

$$= \prod_{t=1}^T p_{\theta}(z_t | z_{t-1}, \tilde{d}_t) p_{\theta}(x_t | z_t, \tilde{d}_t)$$

به منظور آموزش پارامترهای مدل مشابه حالت قبل، کران پایین مشاهدات را به صورت زیر می نویسیم:

$$\log p_{\theta}(x_{1:T} | u_{1:T}, d_0) \quad \text{رابطه}$$

$$\geq E_q \left( \log \frac{p_{\theta}(x_{1:T}, z_{1:T} | u_{1:T}, d_0)}{q_{\phi}(z_{1:T} | x_{1:T}, u_{1:T}, d_0)} \right) \quad (30)$$

$$= F(\theta, \phi)$$

به منظور آموزش شبکه نیاز است تا یک توزیع تغییراتی مناسب انتخاب شود که مانند بخش های قبلی از ساختار توزیع احتمال موخر به صورت مستقیم استفاده می کنیم. نکته مهم درباره این ساختار این است که چون وابستگی بین متغیرهای تصادفی وجود دارد، به منظور تولید نمونه های تصادفی نیازمند پردازش گام های زمانی آینده خواهیم بود. به بیان ریاضی خواهیم داشت:

$$p_{\theta}(z_{1:T} | x_{1:T}, u_{1:T}, d_0) \quad (31)$$

$$= \prod_{t=1}^T p_{\theta}(z_t | z_{t-1}, d_{t:T})$$

در نتیجه توزیع تغییراتی نیز دارای ساختار مشابه به صورت زیر خواهد بود:

$$q_{\phi}(z_{1:T} | x_{1:T}, u_{1:T}, d_0) \quad (32)$$

$$= \prod_{t=1}^T q_{\phi}(z_t | z_{t-1}, \tilde{d}_{t:T})$$

که همانند مدل های فضای حالت غیرخطی برای حل مشکل وابستگی به آینده از روش هموارساز کالمن عمیق باید استفاده کرد. با این تعریف کران پایین مشاهدات مطابق رابطه زیر خواهد بود:

$$F(\theta, \phi) \quad (33)$$

$$= \sum_{t=1}^T E_{q_{\phi}} [\log p_{\theta}(x_t | z_t, \tilde{d}_t)]$$

$$- D_{kl} (q_{\phi}(z_t | z_{t-1}, \tilde{d}_{t:T}) || p_{\theta}(z_t | z_{t-1}, \tilde{d}_t))$$

که تمامی ترم های توزیع شرطی موجود در این معادله از یک توزیع گوسی پیروی می کند که پارامترهای آن توسط یک شبکه عصبی تعیین می شود. ساختار دیگری در همین راستا با نامی مشابه وجود دارد که عملکرد متفاوتی دارد. بر اساس این ساختار یک مکانیزم پنهان وجود دارد

$$F(\theta, \phi) \quad (27)$$

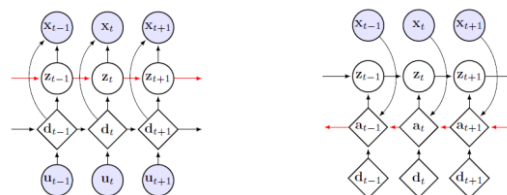
$$= \sum_{t=1}^T E_{q_{\phi}} [\log p_{\theta}(x_t | z_t, d_t)]$$

$$- D_{kl} (q_{\phi}(z_t | x_t, \tilde{d}_t) || p_{\theta}(z_t | \tilde{d}_t))$$

تابع هزینه مناسب برای آموزش این مدل، مشخصاً شباهت نزدیک با تابع هزینه و مکانیزم آموزش خود رمزنگار تغییراتی دارد که نتیجتاً به سادگی و با استفاده از الگوریتم گرادینان نزولی قابل بهینه سازی است. تفاوت تابع هزینه این ساختار با ساختار قبل، وابستگی خروجی به ویژگی های تصادفی استخراج شده است که برای ساخت متغیرهای حالت شبکه عصبی بازگشتی نیز لازم است تا متغیرهای تصادفی مقداردهی اولیه داشته باشند. تمامی ترم های توزیع شرطی موجود در این معادله از یک توزیع گوسی پیروی می کند که پارامترهای آن توسط یک شبکه عصبی تعیین می شود. ایراد اساسی در این ساختار، این است که مدل قادر به فراگرفتن عدم قطعیت های مدلسازی به صورت کامل نیست. برای حل این مشکل ساختار جدیدی به نام شبکه عصبی بازگشتی تصادفی<sup>۱</sup> پیشنهاد شد که متغیرهای تصادفی یک مدل مارکوف مرتبه اول را مدل می کنند [۷۳]. مدل مشابه دیگری نیز با عنوان شبکه تصادفی بازگشتی<sup>۲</sup> در [۷۴] معرفی شده است که به منظور اغای ویژگی های استخراج شده در یک شبکه عصبی بازگشتی به کار گرفته شده است.

### ۳-۳- شبکه عصبی بازگشتی تصادفی

به منظور ایجاد یک نوآوری واریانس بین متغیرهای تصادفی تعریف شده در ساختارهای ارائه شده نیاز به وجود ارتباط بین این ویژگی های تصادفی وجود دارد [۵۵]. در ارتباط غیرمستقیم هر ویژگی تصادفی استخراج شده بر روی ویژگی قطعی در گام بعدی اثر می گذارد در حالی که در ارتباط مستقیم، هر ویژگی تصادفی به صورت مستقیم بر روی ویژگی تصادفی بعد از خود اثرگذار است [۷۳]. شکل ۱۱ ساختار شبکه عصبی بازگشتی تصادفی را به نمایش کشیده است.



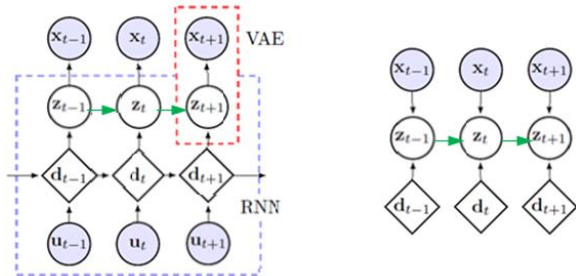
شکل ۱۱ - شبکه های مولد و بازشناسایی ساختار عصبی بازگشتی تصادفی [۱۷]

روابط مربوط به شبکه مولد برای این ساختار به قرار زیر است:

<sup>2</sup> Stochastic recurrent network

<sup>1</sup> Stochastic RNN

ارائه شده فضای حالت در [۱۸] است با این تفاوت که ساختار ساختار شبکه عصبی بازگشتی - خودرمنگار تغییراتی را دستخوش تغییر کرده است. به بیان بهتر، لایه پنهان دارای ساختار است و از یک فرآیند مارکوف مرتبه اول مطابق شکل ۱۲، پیروی می کند.



شکل ۱۲ - مدل شبکه عصبی بازگشتی - خودرمنگار تغییراتی ساختار یافته

در مقایسه با مدل ارائه شده در شکل ۱۱، ارتباط بین متغیرهای پنهان با رنگ سبز به ساختار اضافه شده است. دلیل تغییر دادن این ساختار، حفظ اطلاعات مربوط به بخش نگاشت تصادفی است که در هر گام زمانی باید لحاظ شود. همچنین این ساختار تطبیق پذیری بیش تری با فرم فضای حالت سیستم های کنترلی دارد که در آن، متغیرهای حالت در هر گام زمانی تابعی معین از متغیرهای حالت در زمان های پیشین است که تحت تحریک عدم قطعیت گوسی جمع شوند قرار گرفته است. چون هر متغیر حالت تحت تاثیر عدم قطعیت است، این اثر به صورت غیرخطی باید به حالت بعدی منتقل شود. دلایل مطرح شده برای تغییر ساختار این مدل را می توان عینا برای شبکه عصبی تصادفی نیز به کار برد. نکته بسیار مهم در تغییر ساختار به شکل بیان شده این است که فرآیند آموزش مدل نیازمند استنتاج درباره گام های زمانی آینده نیز خواهد بود که این ارتباط به دلیل مرتبط بودن متغیرهای پنهان به یکدیگر است. برای رفع این مشکل از هموار ساز کالمن عمیق استفاده شده است.

#### ۴-۱- شبیه سازی

در این بخش نتایج حاصل از شبیه سازی ساختارهای معرفی شده و ساختار تغییر یافته را با یکدیگر مقایسه خواهیم کرد. برای این منظور از سه سیستم برای شبیه سازی استفاده شده است که ابتدا در ادامه معرفی خواهند شد.

##### ۴-۱-۱- سیستم گوسی خطی<sup>۵</sup>

رابطه (۳۴) به عنوان اولین سیستم محک که در [۳۳] معرفی شده است مورد ارزیابی قرار می گیرد.

که این مکانیزم به نگاشت معین دنباله ورودی کمک می کند. این ساختار شبکه تصادفی بازگشتی نامیده شده است که جزئیات مربوط به آن در [۷۴] آورده شده است. در مقایسه با مدل ارائه شده، دلیل وابستگی بین ویژگی های قطعی و ویژگی های تصادفی در همان گام زمانی، ویژگی های قطعی دیگر ماهیت قطعی ندارند و دارای یک توزیع احتمال هستند. فرآیند آموزش در این ساختار همانند آنچه که در ساختارهای قبلی ارائه شده است از روش گرادینان نزولی تغییراتی بیز<sup>۱</sup> قابل استفاده می کند.

در تمام ساختارهای ارائه شده ابتدا مدل ویژگی های زمانی داده های ورودی را فرا می گیرد سپس با استفاده از یک ساختار مشابه خودرمنگار تغییراتی یک بازنمایی بعد پایین<sup>۲</sup> از داده ها را برای مدل کردن توزیع احتمال محاسبه می کند. رویکرد عکس این ساختار نیز وجود دارد که عمدتا برای تحلیل داده های ویدیویی به کار گرفته می شود. در این ساختار، مدل سعی می کند ابتدا یک نگاشت در ابعاد پایین از داده ها محاسبه کند و سپس اطلاعات زمانی را فرا بگیرد که با نام خودرمنگار تغییراتی کالمن<sup>۳</sup> شناخته می شود [۷۵]. این ساختار به منظور تشخیص و ردیابی اشیاء موجود در ویدیو<sup>۴</sup> ارائه شده است، از یک خودرمنگار تغییراتی پیچشی<sup>۵</sup> استفاده شده است تا هر فریم تصویر به یک فضای با بعد پایین نگاشت داده شود. سپس هر ویژگی محاسبه شده به یک مدل فضای حالت خطی گوسی<sup>۶</sup> داده می شود که پارامترهای آن توسط ترکیب خطی یک بانک از پارامترهای مدل خطی محاسبه می شود. چون در این حالت، نگاشت ویژگی های هر فریم ورودی و خروجی مشخص است، متغیرهای مخفی موجود در این مدل توسط فیلتر کالمن خطی تخمین زده می شود (چون ورودی و خروجی توسط خود رمنگار تغییراتی محاسبه می شود). کل ساختار کران پایین مشاهدات با روش گرادینان تصادفی تغییراتی بیز آموزش می بیند. از میان ساختارهای موجود، تنها این ساختار در حوزه شناسایی سیستم های غیرخطی ورود نکرده است حال آنکه می تواند برای شناسایی سیستم های چندورودی-چندخروجی<sup>۷</sup> به کار گرفته شود. جزئیات مربوط به این ساختار در [۷۵] آورده شده است.

#### ۴- فضای حالت عمیق ساختار یافته و شبیه سازی مقایسه ای

بر اساس مطلبی که در [۵۵] و [۱۷] بیان شده است، برای یک مدل مولد دنباله نیاز است تا عدم قطعیت به صورت ایستا مدل شود. به بیان بهتر، باید مدل سازی لایه مربوط به نگاشت تصادفی، از گام های زمانی پیش از خود نیز اثر پذیرد تا اطمینان از اینکه سطح مشخصی از عدم قطعیت در مدل سازی لحاظ شده است حاصل گردد. این موضوع دقیقا مطابق با مدل

<sup>۵</sup> Convolutional VAE

<sup>۶</sup> Linear Gaussian state space mode (LGSSM)

<sup>۷</sup> Multivariate systems

<sup>۸</sup> Linear Gaussian system

<sup>۱</sup> Stochastic Gradient Variational Bayes (SGVB)

<sup>۲</sup> Low dimensional manifold

<sup>۳</sup> Kalman Variational Auto-encoder (KVAE)

<sup>۴</sup> Video object detection and tracking

$$\begin{bmatrix} x_{k+1}^1 \\ x_{k+1}^2 \end{bmatrix} = \begin{bmatrix} \left( \frac{x_k^1}{1+(x_k^1)^2} \right) \sin x_k^2 \\ x_k^2 \cos x_k^2 + x_k^1 \exp \left( -\frac{(x_k^1)^2 + (x_k^2)^2}{8} \right) + \frac{u_k^2}{1+u_k^2+0.5 \cos(x_k^1+x_k^2)} \end{bmatrix} \quad (35)$$

$$y_k = \frac{x_k^1}{1+0.5 \sin x_k^2} + \frac{x_k^2}{1+0.5 \sin x_k^1} + e_k \quad (36)$$

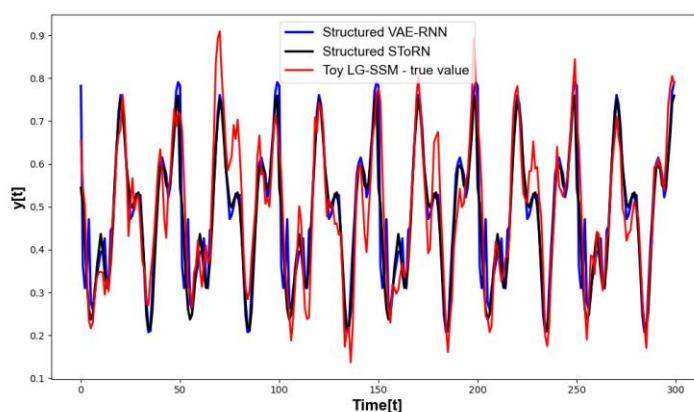
که در آن  $e_k$  نویز اندازه گیری است. لازم بذکر است که در نسخه اصلی این سیستم محک نویز اندازه گیری لحاظ نشده است و نویز اندازه گیری به منظور افزایش پیچیدگی به سیستم اضافه شده است. در شبیه سازی های این بخش، این نویز از یک توزیع گوسی با میانگین ۰ و واریانس ۰.۵ پیروی می کند.

### ۳-۱-۴- سیستم محک Wiener-Hammerstein<sup>3</sup>

فرآیند Wiener-Hammerstein سومین سیستم محکی است که به منظور نمایش کارآیی ساختارهای معرفی شده مورد بررسی قرار گرفته است. داده های مربوط به این فرآیند که دارای نویز فرآیند است توسط پژوهشگران دانشگاه آیندهوون جمع آوری شده است و در وبگاه مربوط به سیستم های محک غیرخطی منتشر شده است.

### ۴-۱-۴- نتایج

در این بخش نتایج مربوط به شناسایی سیستم های معیار معرفی شده در بخش قبل ارائه خواهد شد. کلیه شبیه سازی های صورت گرفته با استفاده از قاعده مونت کارلو<sup>۴</sup> صورت گرفته است. برای این منظور به ازای هر ورودی، از خروجی کدکننده ۱۰۰ نمونه استخراج شده است و خروجی کدگشا میانگین نگاشتی از این ۱۰۰ نمونه است. شکل ۱۳ خروجی مدل را برای سیستم خطی گوسی نشان می دهد.



شکل ۱۳- عملکرد ساختارهای Structured VAE-RNN و Structured STORN برای سیستم خطی گوسی

$$x_{k+1} = \begin{bmatrix} 0.7 & 0.8 \\ 0 & 1 \end{bmatrix} x_k + \begin{bmatrix} -1 \\ 0.1 \end{bmatrix} u_k + w_k \quad (34)$$

$$y_k = [1 \quad 0] x_k + v_k$$

که در آن،  $w_k$  و  $v_k$  به ترتیب نویز فرآیند و نویز اندازه گیری هستند. در شبیه سازی های مربوط به این سیستم، نویزهای فرآیند و اندازه گیری، هر دو از یک توزیع گوسی با میانگین صفر پیروی می کند. واریانس نویز اندازه گیری برابر با ۱ و واریانس نویز فرآیند برابر با ۰.۵ در نظر گرفته شده است. به منظور شناسایی سیستم، برای داده های آموزش و اعتبارسنجی<sup>۱</sup> از یک ورودی که به صورت یکنواخت که در بازه ی ۲.۵- تا ۲.۵ توزیع شده اند، استفاده شده است. هر دو مجموعه از نمونه های آموزش و اعتبارسنجی، بعد از ۵۰ بار اجرا و اعمال عملگر میانگین مورد استفاده قرار گرفته اند. برای داده های تست نیز از ورودی  $u_k = \sin \frac{2k\pi}{10} + \sin \frac{2k\pi}{5}$  استفاده شده است. در مجموعه ۶۰۰۰ داده آموزش و ۲۰۰۰ داده اعتبارسنجی و تست مورد استفاده قرار گرفته است.

### ۲-۱-۴- سیستم محک Narendra-Li<sup>۲</sup>

این سیستم برای اولین بار توسط Narendra معرفی شده است که یک سیستم غیرخطی فرضی است و نمونه فیزیکی آن در دنیای واقعی ساخته نشده است. این سیستم به عنوان دومین سیستم محک برای ارزیابی ساختار ارائه شده در [۳۳] که در حوزه شناسایی سیستم ها در [۳۳] به کار گرفته شده است، مورد ارزیابی قرار گرفته است. معادلات دینامیکی و خروجی این سیستم مطابق رابطه (۳۵) و (۳۶) است.

<sup>3</sup> Wiener-Hammerstein benchmark

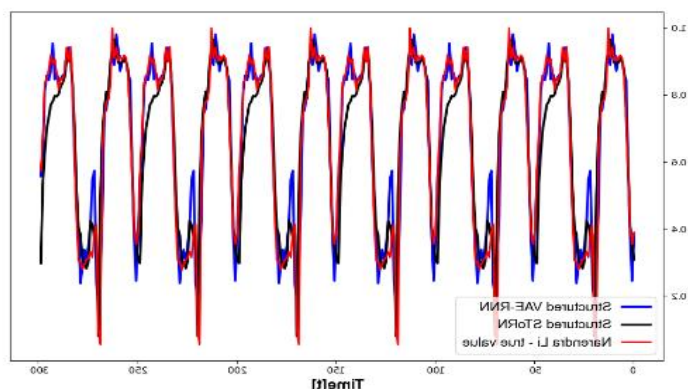
<sup>4</sup> Monte Carlo simulation

<sup>1</sup> Validation

<sup>2</sup> Narendra-Li benchmark

شباهت<sup>۲</sup>  $\frac{1}{T} \sum -\log p(y_t | \mu_{de}, \Sigma_{de})$  نیز برای خودرمنگار تغییراتی  
— شبکه عصبی ساختار یافته به ترتیب برابر با ۱.۰۴ و ۰.۹۹ محاسبه شده  
است که این اعداد برای شبکه تصادفی بازگشتی ساختار یافته برابر با ۰.۹۴  
و ۰.۹۳ است. شکل ۱۴ خروجی مدل های ارائه شده برای سیستم محک  
Narendra-Li را نشان می دهد.

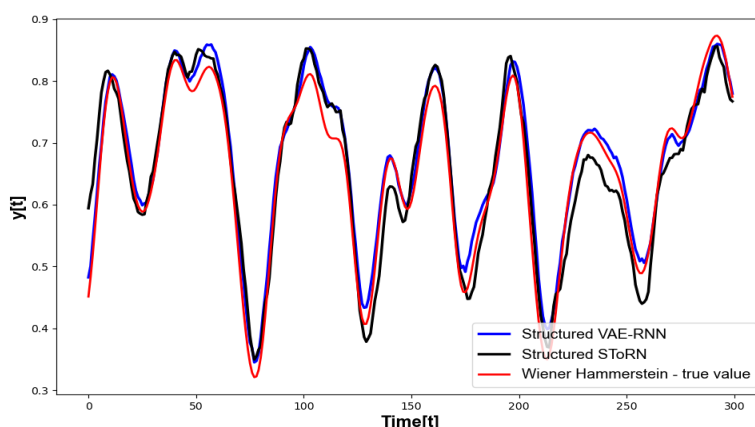
ابعاد لایه پنهان شبکه عصبی بازگشتی برای این شبیه سازی برابر با ۸۰  
و ابعاد لایه تصادفی برابر با ۱۰ در نظر گرفته شده است. همچنین از یک  
لایه بازگشتی برای مدل سازی استفاده شده است. طول دنباله ورودی نیز  
برابر با ۶۰ لحاظ شده است. مقدار موثر خطا<sup>۱</sup> و منفی لگاریتم تابع بیشینه



شکل ۱۴ – عملکرد ساختارهای Structured VAE-RNN و Structured STORN برای سیستم Narendra-Li

شکل ۱۵ نیز خروجی مدل های ارائه شده برای سیستم معیار Wiener-  
Hammerstein را نشان می دهد. برای این حالت نیز از ساختار مشابه با  
دو شبیه سازی قبلی استفاده شده است با این تفاوت که برای این سیستم  
طول دنباله ورودی برابر با ۱۵۰ در نظر گرفته شد. مقدار موثر خطا و تابع  
بیشینه شباهت برای ساختار خودرمنگار تغییراتی — شبکه عصبی بازگشتی  
ساختار یافته به ترتیب برابر با ۰.۶۳ و ۰.۹۱ و برای ساختار شبکه عصبی  
بازگشتی تصادفی ساختار یافته برابر با ۰.۷۴ و ۰.۹۲ محاسبه شده است.

مدل در این حالت ساختاری مشابه با حالت قبل دارد و تنها تفاوت آن  
طول دنباله ورودی است که برای این سیستم برابر با ۵۰ در نظر گرفته شده  
است. همچنین مقدار موثر خطا و تابع بیشینه شباهت برای ساختار  
خودرمنگار تغییراتی — شبکه عصبی بازگشتی ساختار یافته به ترتیب برابر  
با ۰.۷۶ و ۰.۹۲، و برای ساختار شبکه عصبی تصادفی ساختار یافته برابر با  
۱.۰۱ و ۰.۹۶ محاسبه شده است. به عبارتی برای این سیستم عملکرد مدل  
خودرمنگار تغییراتی — شبکه عصبی بازگشتی ساختار یافته بهتر است.



شکل ۱۵ – عملکرد ساختارهای Structured VAE-RNN و Structured STORN برای سیستم Wiener-Hammerstein

<sup>2</sup> Negative log-likelihood

<sup>1</sup> RMS

خلاصه نتایج حاصل شده در جدول ۱ آمده است. نتایج مرتبط به بازگشتی تصادفی و شبکه عصبی بازگشتی تغییراتی از مرجع [۳۳] آورده ساختارهای خودمزنکار تغییراتی - شبکه عصبی بازگشتی، شبکه عصبی شده است.

جدول ۴ - ۱ خلاصه نتایج حاصل شده برای دو ساختار ارائه شده در مقایسه با ساختارهای موجود

| مدل          | مقدار موثر خطا (RMS) |             |                    | منفی لگاریتم تابع شباهت (NLL) |             |                    |
|--------------|----------------------|-------------|--------------------|-------------------------------|-------------|--------------------|
|              | Toy-LGSSM            | Narendra-Li | Wiener-Hammerstein | Toy-LGSSM                     | Narendra-Li | Wiener-Hammerstein |
| VRNN [۳۳]    | ۱/۴۸                 | ۰.۸۹        | ۰.۸۶               | ۱.۸۲                          | ۱.۳۱        | ۱.۳۳               |
| VAE-RNN [۳۳] | ۱/۵۶                 | ۰.۸۴        | ۰.۷۴               | ۱.۹۵                          | ۱.۳۴        | ۱.۲۹               |
| STORN [۳۳]   | ۱/۴۳                 | ۰.۶۴        | ۰.۷۴               | ۱.۷۹                          | ۱.۲۰        | ۱.۰۱               |
| S-VAE-RNN    | ۱/۰۴                 | ۰.۷۶        | ۰.۶۳               | ۰.۹۳                          | ۰.۹۱        | ۰.۹۱               |
| S-STORN      | ۰/۹۴                 | ۱.۰۱        | ۰.۷۴               | ۰.۹۲                          | ۰.۹۲        | ۰.۹۲               |

معرفی شد. نتایج حاصل از شبیه‌سازی نشان می‌دهد که بهره‌وری این مدل-ها قابلیت مقایسه با مدل‌های سنتی حوزه شناسایی سیستم‌ها را داراست، در برخی موارد نتایج بهتر و در برخی موارد نتایج نزدیک گزارش شده است.

## ۶- مراجع

- [1] L. Ljung, "System identification, Theory for the user." System science series, Prentice Hall, Upper Saddle River, NJ, USA, Second edition, 1999.
- [2] Zadeh, L. "On the identification problem." *IRE Transactions on Circuit Theory* 3.4 (1956): 277-281.
- [3] Nelles, Oliver. "Nonlinear system identification, from classical approach to neural networks, fuzzy systems, and Gaussian process". Springer, Berlin, Heidelberg, 2020.
- [4] T. Soderstrom, P. Stocia, "System identification", Prentice-Hall, Inc, 1988.
- [5] Schoukens, Johan, and Lennart Ljung. "Nonlinear system identification: A user-oriented road map." *IEEE Control Systems Magazine* 39.6 (2019): 28-99.
- [6] Billings, Stephen A. "Identification of nonlinear systems—a survey." *IEE Proceedings D (Control Theory and Applications)*. Vol. 127. No. 6. IET Digital Library, 1980.
- [7] Zheng, Qingsheng, and Evangelhos Zafiriou. "Nonlinear system identification for control using Volterra-Laguerre expansion." *Proceedings of 1995 American Control Conference-ACC'95*. Vol. 3. IEEE, 1995.
- [8] Korenberg, Michael J., and Ian W. Hunter. "The identification of nonlinear biological systems:

ساختارهایی که در پژوهش [۳۳] برای شبیه‌سازی انتخاب شده است از جستجوی مشبک<sup>۱</sup> استفاده کرده است حال آنکه برای ساختارهای ارائه شده در این بخش، ساختار شبکه را ثابت در نظر گرفتیم تا بتوانیم دو ساختار را با هم مقایسه کنیم. همچنین برای سیستم Wiener-Hammerstein در مرجع [۳۳] از یک شبکه عصبی بازگشتی ۳ لایه استفاده شده است که به منظور مقایسه ساختارها، در این پژوهش از یک لایه بازگشتی استفاده شده است. بهمین دلیل نتایج ارائه شده در جدول ۱ با نتایج ارائه شده در [۳۳] متفاوت است.

## ۵- نتیجه‌گیری

بدلیل شباهت فرآیندی که در شناسایی سیستم‌ها با مدل‌سازی در حوزه یادگیری ماشین دارد، می‌توان از ساختارها و الگوریتم‌هایی که در حوزه یادگیری ماشین توسعه یافته‌اند، برای هدف شناسایی سیستم‌های غیرخطی استفاده کرد. این انتقال ابزار از حوزه یادگیری ماشین به حوزه شناسایی سیستم‌ها، نه تنها ابزارهای موجود برای مدل‌سازی را توسعه می‌دهد بلکه در برخی موارد می‌تواند از ابزارهای موجود بهتر عمل کند. در این مقاله مروری، ضمن معرفی ابزارها و بررسی مطالعات پژوهشی صورت گرفته در این راستا، مدل‌های متغیر پنهان به عنوان ابزاری با ظرفیت بالا، به منظور مدل‌سازی و شناسایی فضای حالت سیستم‌های غیرخطی معرفی شد. مدل‌های متغیر پنهان از خانواده مدل‌های مولد هستند که یک نگراشت تصادفی از داده‌های ورودی محاسبه می‌کند که تغییرات موجود در داده‌ها را، بر خلاف نگاشت‌های معین، لحاظ می‌کند. در پایان با اعمال تغییر جزئی در ساختار مدل‌های متغیر پنهان به منظور تطبیق‌پذیری بیش‌تر با مفهوم متغیر حالت در سیستم‌های کنترل، مدل ساختار یافته متغیر پنهان

<sup>1</sup> Grid search

- [23] Gianluigi Pillonetto, Aleksandr Aravkin, Daniel Gedon, Lennart Ljung, Antônio H. Ribeiro, Thomas B. Schön, Deep network for system identification, arXiv, 2023.
- [24] Ljung, Lennart, et al. "Deep learning and system identification." *IFAC-PapersOnLine* 53.2 (2020): 1175-1181.
- [25] M. Forgiione and D. Piga. Dynonet: A neural network architecture for learning dynamical systems. *Int. J. Adapt. Control Signal Process.*, 35(4):612–626, 2021.
- [26] J. Hendriks, F.K. Gustafsson, A.H. Ribeiro, A. Wills, and T.B. Schon. Deep energy-based NARX models. In *Proceedings of the 19th IFAC Symposium on System Identification (SYSID)*, 2021.
- [27] LeCun, Yann, et al. "Energy-based models in document recognition and computer vision." *Ninth International Conference on Document Analysis and Recognition (ICDAR 2007)*. Vol. 1. IEEE, 2007.
- [28] Andersson, Carl, et al. "Deep convolutional networks in system identification." *2019 IEEE 58th conference on decision and control (CDC)*. IEEE, 2019.
- [29] Lea, Colin, et al. "Temporal convolutional networks: A unified approach to action segmentation." *Computer Vision–ECCV 2016 Workshops: Amsterdam, The Netherlands, October 8–10 and 15–16, 2016, Proceedings, Part III* 14. Springer International Publishing, 2016.
- [30] R. Calandra, J. Peters, C. Rasmussen, and M.P. Deisenroth. Manifold gaussian processes for regression. In *2016 International Joint Conference on Neural Networks (IJCNN)*, pages 3338–3345, 2016.
- [31] Y. Cho and L. Saul. Kernel methods for deep learning. In *Advances in Neural Information Processing Systems*, volume 22, 2009.
- [32] Nagel, Tobias, and Marco F. Huber. "Autoencoder-inspired Identification of LTI systems." *2021 European Control Conference (ECC)*. IEEE, 2021.
- [33] D. Gedon, N. Wahlström, T.B. Schön, and L. Ljung. Deep state space models for nonlinear system identification. In *Proceedings of the 19th IFAC Symposium on System Identification (SYSID)*, 2021.
- Wiener kernel approaches." *Annals of Biomedical Engineering* 18 (1990): 629-654.
- [9] Schoukens, Maarten, and Koen Tiels. "Identification of block-oriented nonlinear systems starting from linear approximations: A survey." *Automatica* 85 (2017): 272-292.
- [10] Chiuso, Alessandro, and Gianluigi Pillonetto. "System identification: A machine learning perspective." *Annual Review of Control, Robotics, and Autonomous Systems* 2 (2019): 281-304.
- [11] Pillonetto, Gianluigi, et al. "Kernel methods in system identification, machine learning and function estimation: A survey." *Automatica* 50.3 (2014): 657-682.
- [12] Bishop, Christopher M., and Nasser M. Nasrabadi. *Pattern recognition and machine learning*. Vol. 4. No. 4. New York: springer, 2006.
- [13] Heaton, Jeff. Ian goodfellow, yoshua bengio, and aaron courville: "Deep learning." (2018): 305-307.
- [14] Bengio, Yoshua, Aaron Courville, and Pascal Vincent. "Representation learning: A review and new perspectives." *IEEE transactions on pattern analysis and machine intelligence* 35.8 (2013): 1798-1828.
- [15] Noroozi, Mehdi, and Paolo Favaro. "Unsupervised learning of visual representations by solving jigsaw puzzles." *European conference on computer vision*. Cham: Springer International Publishing, 2016.
- [16] Kingma, Diederik P., and Max Welling. "Auto-encoding variational bayes." *arXiv preprint arXiv: 1312.6114* (2013).
- [17] Fraccaro, Marco. "Deep latent variable models for sequential data." *English. PhD thesis* DTU University (2018).
- [18] Barber, David. *Bayesian reasoning and machine learning*. Cambridge University Press, 2012.
- [19] Koller, Daphne, and Nir Friedman. *Probabilistic graphical models: principles and techniques*. MIT press, 2009.
- [20] Hinton, Geoffrey E. "A practical guide to training restricted Boltzmann machines." *Neural Networks: Tricks of the Trade: Second Edition*. Berlin, Heidelberg: Springer Berlin Heidelberg, 2012. 599-619.
- [21] Fischer, Asja, and Christian Igel. "An introduction to restricted Boltzmann machines." *Progress in Pattern Recognition, Image Analysis, Computer Vision, and Applications: 17th Iberoamerican Congress, CIARP 2012, Buenos Aires, Argentina, September 3-6, 2012. Proceedings* 17. Springer Berlin Heidelberg, 2012.
- [22] Raia Hadsell, Sumit Chopra, and Yann LeCun. Dimensionality reduction by learning an invariant mapping. In *CVPR*, 2006.

- [47] Liang, Tailin, et al. "Pruning and quantization for deep neural network acceleration: A survey." *Neurocomputing* 461 (2021): 370-403.
- [48] Frankle, Jonathan, and Michael Carbin. "The lottery ticket hypothesis: Finding sparse, trainable neural networks." *arXiv preprint arXiv: 1803.03635* (2018).
- [49] Hinton, Geoffrey, Oriol Vinyals, and Jeff Dean. "Distilling the knowledge in a neural network." *arXiv preprint arXiv: 1503.02531* (2015).
- [50] Gou, Jianping, et al. "Knowledge distillation: A survey." *International Journal of Computer Vision* 129 (2021): 1789-1819.
- [51] Hastie, Trevor, et al. *The elements of statistical learning: data mining, inference, and prediction*. Vol. 2. New York: springer, 2009.
- [52] Doersch, Carl. "Tutorial on variational autoencoders." *arXiv preprint arXiv: 1606.05908* (2016).
- [53] Burda, Yuri, Roger Grosse, and Ruslan Salakhutdinov. "Importance weighted autoencoders." *arXiv preprint arXiv: 1509.00519* (2015) [40] Barber, David, A. Taylan Cemgil, and Silvia Chiappa, eds. *Bayesian time series models*. Cambridge University Press, 2011.
- [54] Makhzani, Alireza, et al. "Adversarial autoencoders." *arXiv preprint arXiv: 1511.05644* (2015). [39] Burda, Yuri, Roger Grosse, and Ruslan Salakhutdinov. "Importance weighted autoencoders." *arXiv preprint arXiv: 1509.00519* (2015)
- [55] Girin, Laurent, et al. "Dynamical variational autoencoders: A comprehensive review." *arXiv preprint arXiv:2008.12595* (2020).
- [56] Beal, Matthew James. *Variational algorithms for approximate Bayesian inference*. University of London, University College London (United Kingdom), 2003.
- [57] Hochreiter, Sepp, and Jürgen Schmidhuber. "Long short-term memory." *Neural computation* 9.8 (1997): 1735-1780.
- [58] Chung, Junyoung, et al. "Empirical evaluation of gated recurrent neural networks on sequence modeling." *arXiv preprint arXiv:1412.3555* (2014).
- [59] Su, Yuanhang, and C-C. Jay Kuo. "Recurrent neural networks and their memory behavior: a survey." *APSIPA Transactions on Signal and Information Processing* 11.1 (2022).
- [60] Graves, Alex. "Generating sequences with recurrent neural networks." *arXiv preprint arXiv:1308.0850* (2013).
- [34] M. Karl, M. Soelch, J. Bayer, and P. van der Smagt. Deep variational Bayes filters: Unsupervised learning of state space models from raw data, 2017.
- [35] M. Watter, J. Springenberg, J. Tobias, J. Boedecker, and M. Riedmiller. Embed to control: A locally linear latent dynamics model for control from raw images. In Proceedings of the 28<sup>th</sup> International Conference on Neural Information Processing Systems, Volume 2, pages 2746–2754, Cambridge, MA, USA, 2015. MIT Press.
- [36] Rangapuram, Syama Sundar, et al. "Deep state space models for time series forecasting." *Advances in neural information processing systems* 31 (2018).
- [37] Courts, Jarrad, et al. "Variational state and parameter estimation." *IFAC-PapersOnLine* 54.7 (2021): 732-737.
- [38] Courts, Jarrad, et al. "Variational System Identification for Nonlinear State-Space Models." *arXiv preprint arXiv: 2012.05072* (2020).
- [39] Menghani, Gaurav. "Efficient deep learning: A survey on making deep learning models smaller, faster, and better." *ACM Computing Surveys* 55.12 (2023): 1-37.
- [40] Xu, Canwen, and Julian McAuley. "A survey on model compression and acceleration for pretrained language models." *Proceedings of the AAAI Conference on Artificial Intelligence*. Vol. 37. No. 9. 2023.
- [41] Choudhary, Tejalal, et al. "A comprehensive survey on model compression and acceleration." *Artificial Intelligence Review* 53 (2020): 5113-5155.
- [42] Cheng, Yu, et al. "A survey of model compression and acceleration for deep neural networks." *arXiv preprint arXiv:1710.09282* (2017).
- [43] Cheng, Yu, et al. "Model compression and acceleration for deep neural networks: The principles, progress, and challenges." *IEEE Signal Processing Magazine* 35.1 (2018): 126-136.
- [44] Liang, Tailin, et al. "Pruning and quantization for deep neural network acceleration: A survey." *Neurocomputing* 461 (2021): 370-403.
- [45] Rokh, Babak, Ali Azarpeyvand, and Alireza Khanteymoori. "A comprehensive survey on model quantization for deep neural networks." *arXiv preprint arXiv: 2205.07877* (2022).
- [46] Gholami, Amir, et al. "A survey of quantization methods for efficient neural network inference." *Low-Power Computer Vision*. Chapman and Hall/CRC, 2022. 291-326.

- [61] Krishnan, Rahul G., Uri Shalit, and David Sontag. "Deep kalman filters." *arXiv preprint arXiv: 1511.05121* (2015).
- [62] Bahdanau, Dzmitry, Kyunghyun Cho, and Yoshua Bengio. "Neural machine translation by jointly learning to align and translate." *arXiv preprint arXiv: 1409.0473* (2014).
- [63] Vaswani, Ashish, et al. "Attention is all you need." *Advances in neural information processing systems* 30 (2017).
- [64] Han, Kai, et al. "A survey on vision transformer." *IEEE transactions on pattern analysis and machine intelligence* 45.1 (2022): 87-110.
- [65] Khan, Salman, et al. "Transformers in vision: A survey." *ACM computing surveys (CSUR)* 54.10s (2022): 1-41.
- [66] Liu, Yang, et al. "A survey of visual transformers." *IEEE Transactions on Neural Networks and Learning Systems* (2023).
- [67] Lin, Tianyang, et al. "A survey of transformers." *AI Open* (2022).
- [68] Rosendahl, Jan, et al. "Analysis of positional encodings for neural machine translation." *Proceedings of the 16th International Conference on Spoken Language Translation*. 2019.
- [69] Beintema, Gerben I., Maarten Schoukens, and Roland Tóth. "Deep subspace encoders for nonlinear system identification." *Automatica* 156 (2023): 111210.
- [70] Masti, Daniele, and Alberto Bemporad. "Learning nonlinear state-space models using autoencoders." *Automatica* 129 (2021): 109666.
- [71] Lopez, Ryan, and Paul J. Atzberger. "Variational autoencoders for learning nonlinear dynamics of physical systems." *arXiv preprint arXiv:2012.03448* (2020).
- [72] Chung, Junyoung, et al. "A recurrent latent variable model for sequential data." *Advances in neural information processing systems* 28 (2015).
- [73] Bayer, Justin, and Christian Osendorfer. "Learning stochastic recurrent networks." *arXiv preprint arXiv: 1411.7610* (2014).
- [74] Fraccaro, Marco, et al. "Sequential neural models with stochastic layers." *Advances in neural information processing systems* 29 (2016).
- [75] Fraccaro, Marco, et al. "A disentangled recognition and nonlinear dynamics model for unsupervised learning." *Advances in neural information processing systems* 30 (2017).