تولید درخت‌بانک سازه‌ای زبان فارسی به روش تبدیل خودکار

محمدحسین دهقان* **و هشام فیلی**

دانشکدهٔ مهندسی برق و کامپیوتر، پردیس دانشکده‌های فنی، دانشگاه تهران، تهران، ایران

چکیده

درخت‌بانک از مهم‌ترین و پرکاربردترین منابع مورد استفاده در زمینهٔ پردازش زبان طبیعی است. دو نوع از پرکاربردترین درخت‌بانک‌ها، درخت‌بانک وابستگی و درخت‌بانک سازه‌ای است. با توجه به نبود درخت‌بانک سازه‌ای با حجم بزرگ در زبان فارسی در این مقاله به بررسی روشی ارائه‌شده در تبدیل درخت‌بانک وابستگی به سازه‌ای می‌پردازیم. سپس مشکلات این روش را در زبان فارسی و انگلیسی بررسی و با ارائهٔ راه‌کارهایی کیفیت تبدیل را بهبود می‌بخشیم. نخستین راه‌کار، تصحیح مکان اتصال سازه‌ها در درخت سازه‌ای به‌ازای هر رابطهٔ وابستگی است. راه‌کار دوم، انجام مکاشفه‌ای به‌صورت پس‌پردازش و بر روی خروجیِ ساختار سازه‌ای این روش است که، کیفیت نهایی درخت‌های سازه‌ای را بهبود می‌بخشد. نتایج حاصل از آزمایش‌ها نشان می‌دهد که، روش تبدیل با کمک راه‌کارهای ارائه‌شده حدود ۲۵/۸۵ درصد در زبان فارسی و ۴/۳۹ درصد در زبان انگلیسی دارای کیفیت بالاتری نسبت به حالتی است که از راه‌کارهای پیشنهادی استفاده نشود. در ادامه با کمک روش تبدیل و درخت‌بانک وابستگی موجود در زبان فارسی، یک درخت بانک سازه‌ای تولید کرده و به کمک آن تجزیه‌گری سازه‌ای را آموزش داده‌ایم. کیفیت تجزیه‌گر آموزش‌داده‌شده با استفاده از درخت‌بانک حاصل از روش تبدیل و راه‌کارهای پیشنهادی این پژوهش نسبت به‌حالتی که از راه‌کارهای پیشنهادی استفاده نشود، بهبود ۲۱ درصدی را نشان می‌دهد.

واژگان کلیدی: پردازش زبان طبیعی، زبان فارسی، درخت بانک وابستگی، درخت بانک سازه‌ای، تجزیه‌گر سازه‌ای

۱- مقدّمه

پردازش زبان طبیعی[1] یکی از زیرشاخه‌های هوش مصنوعی، زبان‌شناسی و علوم رایانه است؛ که کاربردهای فراوانی در پردازش زبان گفتار و نوشتار دارد (کومار، ۲۰۱۱). از جمله موارد پیچیده پردازش زبان طبیعی می‌توان به ترجمه ماشینی و پاسخ‌دادن به پرسش‌ها اشاره کرد. یکی از منابع پراستفاده در پردازش زبان طبیعی، درخت‌بانک[2] است. درخت‌بانک مجموعه‌ای از جملات است که براساس یک نظریهٔ زبانی تجزیه شده‌اند. به‌عنوان نمونه می‌توان به درخت‌بانک وابستگی (حجیک، ۱۹۹۸)، درخت‌بانک سازه‌ای[3] (مارکوس و همکاران، ۱۹۹۳) و درخت‌بانک دستور ساخت سازه‌ای هسته بنیان[4](اپن و همکاران، ۲۰۰۲) اشاره کرد.

تولید درخت بانک به‌صورت دستی کاری هزینه‌بر و در عین حال زمان‌بر است. علاوه‌براین حتی در صورت وجود درخت بانک‌های مختلف در یک زبان، برخی کاربردهای پردازش زبان طبیعی نیاز به درخت بانک‌های معادل یکدیگر دارد (مک‌دونلاد و همکاران، ۲۰۰۶). به‌همین دلیل نیاز است که درخت‌بانک از نوعی به نوع دیگر تبدیل شود. تبدیل خودکار درخت‌بانک‌ها به یکدیگر کمک می‌کند که علاوه‌بر صرفه‌جویی در زمان و هزینه، بتوان برای درخت‌بانک‌های موجود در یک زبان، درخت‌بانک معادل آن را به‌دست‌آورد. از این‌رو روش‌های تبدیل خودکار درخت‌بانک‌ها مورد توجه قرار گرفته است و راه‌کارهای

[1] Natural Language Processing
[2] Treebank
[3] Phrase structure
[4] Head-driven Phrase Structure Grammar

مختلفی برای تبدیل انواع درخت‌بانک‌ها به یکدیگر نظیر وابستگی به سازه‌ای و برعکس به وجود آمده است.

برخی از انواع درخت‌بانک‌ها کاربرد بیشتری نسبت به انواع دیگر دارند. به‌عنوان مثال دو نمونه از پرکاربردترین انواع درخت‌بانک‌ها، درخت بانک وابستگی[1] مانند درخت بانک پراگ در زبان چکی[2] (حجیک، ۱۹۹۸) و درخت‌بانک سازه‌ای[3] مانند درخت‌بانک پن در زبان انگلیسی[4] (مارکوس و همکاران، ۱۹۹۳) است.

با توجه به اهمیت این دو نوع درخت‌بانک راه‌کارهای فراوانی در جهت تبدیل این درخت‌بانک‌ها به یکدیگر ارائه شده است. در زبان فارسی نیز دو نوع درخت‌بانک وابستگی و سازه‌ای موجود است، که متأسفانه حجم درخت بانک سازه‌ای کوچک است.

Perdt درخت‌بانک به قالب وابستگی و دارای حدود سی‌هزار جمله از متون مختلف معاصر فارسی است. این درخت‌بانک، وابستگی بین کلمات را در جمله به قالب CoNLL نمایش می‌دهد (رسولی و همکاران، ۲۰۱۳).

PerTreebank درخت‌بانکی شامل ۱۰۲۸ جملهٔ انتخاب‌شده از پیکرهٔ[5] بی‌جن‌خان[6] است که دارای نمایش درختی به قالب دستور ساخت‌سازه‌ای هسته‌بنیان است. این درخت‌بانک اولین درخت بانک تهیه‌شده برای زبان فارسی به قالب دستور ساخت‌سازه‌ای هسته‌بنیان است (قیومی، ۲۰۱۲a). دستور ساخت‌سازه‌ای هسته‌بنیان شکل پیشرفتهٔ دستور ساخت‌سازه‌ای است که توسط *چامسکی* معرفی شده و زیرمجموعهٔ دستور زایشی[7] است. در این نظریه علاوه‌بر ساختار نحوی، تعبیر معنایی نیز وجود دارد (اپن و همکاران، ۲۰۰۲). درخت‌بانک DepPerTreebank به قالب وابستگی و معادل درخت‌بانک PerTreebank است. درخت‌بانک DepPerTreebank نیز همانند درخت‌بانک Perdt به قالب CoNLL نمایش داده شده است و برای مقاصد پژوهشی به رایگان در دسترس قرار دارد (قیومی و کوهن، ۲۰۱۴). تبدیل ساختار سازه‌ای به وابستگی، نسبت به عکس آن ساده‌تر و به‌طورتقریبی قاعده‌مند است و تنها نیاز به یک جدول برای مشخص‌کردن هستهٔ هر سازه دارد. تلاش برای این تبدیل به‌صورت قاعده‌مند در (ژیا و پالمر، ۲۰۰۱) انجام شده است. در (قیومی و کوهن، ۲۰۱۴) نیز از یک روش قاعده‌مند جهت تبدیل PerTreebank به DepPerTreebank استفاده شده است.

نخستین تلاش‌ها برای تبدیل ساختار وابستگی به سازه‌ای نیز به‌صورت قاعده‌مند بوده و از اطلاعات آماری و دیگر اطلاعات موجود در ساختار وابستگی نظیر تعداد فرزندان یک هسته و نوع وابستگی[8] استفاده نشده است. در (کاوینگتن، ۱۹۹۴) با کمک نظریهٔ ایکس تیره[9] و فرضیاتی ساده درخت سازه‌ای به‌طور مستقیم از درخت وابستگی به‌دست می‌آید. عدم استفاده از اطلاعات آماری و دیگر اطلاعات موجود در درخت‌بانک، موجب کاهش دقت[10] در این روش تبدیل می‌شود. بر پایهٔ این نظریه روش‌های تبدیل گسترش یافتند و با افزودن جداول و تغییر فرضیات، نتایج بهبود داده شد؛ اما عمده‌ترین مشکل این روش‌ها یعنی عدم استفاده از اطلاعات آماری سبب محدودیت این دسته از روش‌ها شده است. دو مورد دیگر از روش‌های اصلی بر پایهٔ این نظریه نیز در (کالینز و همکاران، ۱۹۹۹) و (ژیا و پالمر، ۲۰۰۱) آمده است. روش ارائه شده در (کاوینگتن، ۱۹۹۴) دارای دقت کم در مقابل فراخوانی[11] بالا در هنگام عملیات تبدیل است. (کالینز و همکاران، ۱۹۹۹) برای به‌دست‌آوردن دقت بالاتر تلاش کرده‌اند؛ ولی افزایش دقت همراه با کاهش فراخوانی بوده است. درنهایت (ژیا و پالمر، ۲۰۰۱) با اضافه‌کردن سه جدول که در جهت تعیین انشعاب[12] برای هر گرهٔ داخلی[13] در ساختار سازه‌ای است، سعی در ایجاد تعادلی بین دقت و فراخوانی کرده‌اند.

(ژیا و همکاران، ۲۰۰۹) با استفاده از اطلاعات آماری موجود در ساختار وابستگی و سازه‌ای و همچنین برخی اطلاعات زبانی، الگوریتم دومرحله‌ای ارائه کرده‌اند. در مرحلهٔ نخست که مرحلهٔ آموزش مدل است دو درخت‌بانک وابستگی و سازه‌ای معادل به‌عنوان ورودی دریافت و قوانین تبدیل[14] برای هر یال وابستگی با کمک آنها استخراج می‌شود. قوانین تبدیل درحقیقت زیردرخت دوتایی[15] است که به‌ازای هر یال وابستگی و به‌کمک درخت سازه‌ای معادل آن به‌دست‌می‌آید. به‌ازای هر یال وابستگی در درخت وابستگی، یک قانون تبدیل به‌دست می‌آید، اما به‌دلیل

[1] Dependency treebank
[2] Prague dependency treebank
[3] Constituency treebank
[4] Penn English Treebank
[5] Corpus
[6] http://ece.ut.ac.ir/dbrg/bijankhan/
[7] Generative Grammar

[8] Dependency type
[9] X-bar theory
[10] Precision
[11] Recall
[12] Project
[13] Internal node
[14] Conversion rule
[15] Binary

وجود درخت‌های دیگر در درخت‌بانک، در مجموعهٔ نهایی قوانین تبدیل ممکن است، به‌ازای یک یال وابستگی چندین قانون تبدیل به‌دست آید. پس از تکمیل مرحلهٔ آموزش، مجموعه‌ای شامل یال‌های وابستگی و قوانین تبدیل معادل آن‌ها به‌دست می‌آید، که در مرحلهٔ دوم قابل استفاده است. به‌دلیل وجود ابهام در قوانین تبدیل، برخی اطلاعات دیگر موجود در ساختار وابستگی نظیر نوع وابستگی، وجود وابسته بین هسته و وابسته، نیز ذخیره می‌شود. در مرحلهٔ دوم الگوریتم، درخت وابستگی به‌عنوان ورودی الگوریتم دریافت می‌شود. در این مرحله به‌ازای هر یال وابستگی به‌کمک مجموعه قوانین تبدیل به‌دست‌آمده در مرحلهٔ نخست، درخت دوتایی سازه‌ای معادل آن به‌دست‌می‌آید. درنهایت با اتصال زیردرخت‌های سازه‌ای به یکدیگر درخت سازه‌ای معادل درخت وابستگی ورودی ساخته می‌شود.

در این مقاله به پیاده‌سازی الگوریتم تبدیل درخت‌بانک وابستگی به سازه‌ای ارائه‌شده در (ژیا و همکاران، ۲۰۰۹) بر روی فارسی و انگلیسی می‌پردازیم. پس از بررسی کامل خروجی الگوریتم و با استفاده از نتایج به‌دست‌آمده، خطاهای رخ داده در خروجی الگوریتم را دسته‌بندی می‌کنیم. علت وقوع هریک از خطاها را بررسی و برای برخی مشکل‌های موجود در خروجی الگوریتم، راه‌کارهایی با استفاده از مکاشفه و یادگیری ماشین با کمک طبقه‌بند ارائه می‌کنیم. به‌طور خلاصه نوآوری‌های مقاله به شرح زیر است:

- مشکلات موجود در این الگوریتم را بر روی زبان فارسی و انگلیسی مورد بررسی قرار داده و دسته‌بندی می‌کنیم.
- با ارائهٔ راه‌کاری آماری و با استفاده از طبقه‌بند، به اصلاح روش و بهبود نتیجهٔ خروجی می‌پردازیم.
- با پس‌پردازش خروجی حاصل از الگوریتم و اعتبارسنجی بر روی گره‌های سازه‌ای تکی[1]، درخت سازه‌ای حاصل از خروجی الگوریتم را اصلاح می‌کنیم.
- با کمک تبدیل درخت‌بانک وابستگی Perdt، درخت‌بانک سازه‌ای برای زبان فارسی به‌دست‌می‌آوریم.
- درخت بانک به‌دست‌آمده را به‌کمک آموزش تجزیه‌گر سازه‌ای[2] مورد ارزیابی قرار می‌دهیم.

برای انجام آزمایش‌ها در زبان انگلیسی از درخت‌بانک پن و برای زبان فارسی از درخت‌بانک سازه‌ای PerTreeBank و درخت‌بانک وابستگی DepPerTreeBank که معادل درخت‌بانک سازه‌ای PerTreeBank است، استفاده می‌کنیم. به‌منظور مقایسهٔ نتایج به‌دست‌آمده در این مقاله با کارهای پیشین از روش ارائه‌شده در مقاله (ژیا و پالمر، ۲۰۰۱) در جهت تبدیل ساختار سازه‌ای به وابستگی در زبان انگلیسی استفاده می‌کنیم.

در این مقاله به بررسی کارهای مرتبط در بخش ۲ پرداخته؛ سپس الگوریتم ارائه‌شده در جهت تبدیل ساختار وابستگی به سازه‌ای، در بخش ۳ توضیح داده می‌شود. نتایج حاصل از الگوریتم تبدیل بر روی دو زبان فارسی و انگلیسی در بخش ۴ بررسی شده و در ادامه در بخش‌های ۵ و ۶ به‌ترتیب به توضیح تجزیه‌گر سازه‌ای آموزش‌داده‌شده و نتیجه‌گیری و کارهای آینده می‌پردازیم.

۲- مروری بر کارهای پیشین

تلاش‌های زیادی در جهت تبدیل دو ساختار سازه‌ای و وابستگی به یکدیگر انجام شده است. هر دو ساختار وابستگی و سازه‌ای نمایش درختی از یک جمله هستند. در ساختار وابستگی هر گره در درخت نمایش‌دهندهٔ یک کلمه است و رابطهٔ وابستگی بین کلمات را به‌طور مستقیم نشان می‌دهد. در ساختار سازه‌ای گره‌های خارجی (برگ‌ها) نمایش‌دهندهٔ کلمات و یال‌ها نمایش دهندهٔ انشعاب‌ها هستند. درحالی‌که در ساختار وابستگی نمی‌توان انشعاب را به‌طور مستقیم به‌دست‌آورد، در ساختار سازه‌ای نیز نمی‌توان رابطهٔ وابستگی بین کلمات را به‌طور مستقیم به‌دست‌آورد. دو ساختار می‌توانند دارای ویژگی‌های مشترکی نیز باشند. به‌عنوان نمونه ممکن است هر دو ساختار به‌صورت افکنشی[3] یا غیرافکنشی[4] باشند (قیومی و کوهن، ۲۰۱۴).

تبدیل ساختار سازه‌ای به وابستگی با کمک جدولی که هسته هر سازه را مشخص می‌کند، امکان پذیر است. نمونه‌ای از این روش در (ژیا و پالمر ۲۰۰۱) ارائه شده است. این جدول به‌ازای هر برچسب گرهٔ داخلی، مشخص می‌کند که کدام‌یک از فرزندان گرهٔ داخلی باید به‌عنوان هسته انتخاب شود. دو مرحلهٔ اصلی این روش به‌صورت زیر است:

[1] Unary phrase structure link
[2] Phrase structure parser
[3] Projective
[4] Non-projective

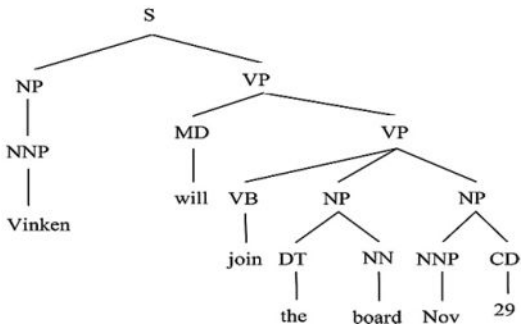
۱- در ساختار سازه‌ای با استفاده از جدول قواعد هسته، هستهٔ هر سازه را از میان فرزندان آن گره انتخاب کن.

۲- در ساختار وابستگی، سایر گره‌های غیرهسته را به‌عنوان وابسته به هستهٔ انتخاب‌شده متصل کن.

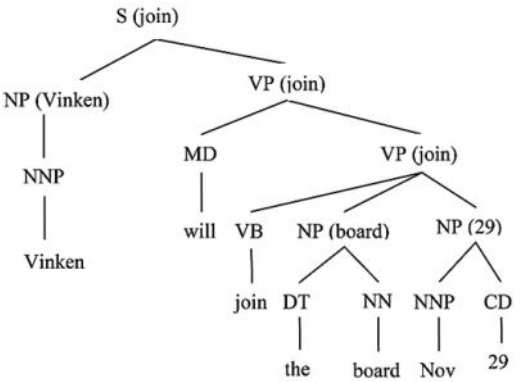
در این روش ابتدا یک درخت سازه‌ای به‌عنوان ورودی دریافت می‌شود؛ سپس با کمک جدول به‌ازای هر زیردرخت، یکی از فرزندان به‌عنوان هسته انتخاب می‌شود. برچسب ریشهٔ زیردرخت در جدول جستجو شده و در هنگام رسیدن به سطری از جدول که اولویت فرزندان را برای ریشهٔ آن زیردرخت نشان می‌دهد، متوقف می‌شود. سپس برچسب فرزند با اولویت بالاتر انتخاب شده، به‌عنوان برچسب گرهٔ پدرش جایگذاری می‌شود؛ که درنتیجه هستهٔ دیگر فرزندان گرهٔ پدر به‌شمار می‌آید. شکل (۱) اولین جمله درخت بانک پن را که دارای ساختار سازه‌ای و شکل (۲) جملهٔ شکل (۱) را که توسط این الگوریتم از پایین به بالا هستهٔ هر سازه را به‌کمک جدول (۱) مشخص کرده است، نشان می‌دهد. الگوریتم به‌ازای هر گره داخلی با توجه به جهت و مجموعه اولویت، یکی از فرزندان را به‌عنوان هسته انتخاب می‌کند. شکل (۳) ساختار وابستگی به‌دست‌آمدهٔ حاصل از شکل (۲) را نمایش داده است. این الگوریتم، یک راه‌کار قاعده‌مند است که نیاز به مرحلهٔ آموزش ندارد و تنها با دراختیارداشتن جدول مذکور اجرای آن امکان‌پذیر است. در پژوهش انجام‌شده توسط (ژیا، ۲۰۰۱) میزان خطا در این روش ناچیز عنوان شده است.

(گویال و کولکرنی، ۲۰۱۴) راه‌کاری جهت تبدیل ساختار سازه‌ای به وابستگی را در زبان سانسکریت که یک زبان بدون ترتیب کلمه[1] است، ارائه دادند. عملیات تبدیل با اضافه‌کردن علائمی به هر زیردرخت در ساختار سازه‌ای، که به‌کمک آن‌ها هستهٔ هر زیردرخت را می‌توان تشخیص داد، صورت می‌گیرد. پس از آن با اتصال هسته‌ها به یکدیگر درخت وابستگی تشکیل می‌شود.

دو راهکار قاعده‌مند جهت تبدیل ساختار وابستگی به سازه‌ای در (کاوینگتن، ۱۹۹۴) و (کالینز و همکاران، ۱۹۹۹) ارائه شده است. هر دو راه‌کار به همراه راه‌کار جدیدی بر روی درخت‌بانک پن در (ژیا و پالمر، ۲۰۰۱) مورد ارزیابی قرار گرفته است. هر سه راه‌کار در استفاده از نظریه ایکس تیره مشترک بوده و به‌صورت قاعده‌مند عمل می‌کنند. دو راه‌کار نخست بدون استفاده از اطلاعات ساختاری عمل می‌کنند، در حالیکه راهکار سوم با بررسی ساختارهای سازه‌ای و وابستگی و ارائه جداولی سعی در تبدیل ساختارها به یکدیگر کرده است.



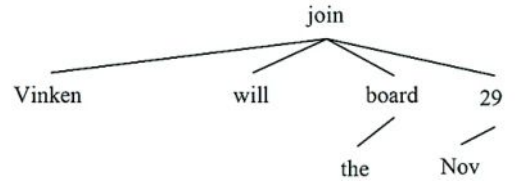
(شکل- ۱): یک نمونه درخت سازه‌ای موجود در پن



(شکل- ۲): برچسب گذاری هسته هر سازه در درخت شکل (۱)

(جدول- ۱): قسمتی از جدول تعیین هسته برای گره‌های داخلی درخت سازه‌ای

مجموعه اولویت	جهت	برچسب گره داخلی
VP/SBAR/S	راست	S
NP/NN/NNS/NNP/CD/DT	راست	NP
VP/VB/VBN/VBZ/VBD	راست	VP

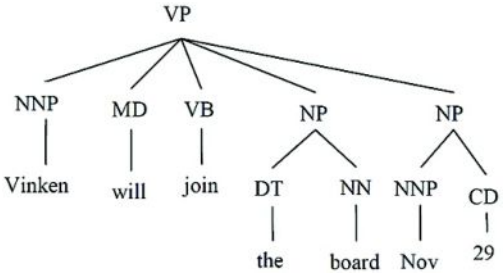


(شکل- ۳): ساختار وابستگی معادل ساختار سازه‌ای شکل (۱)

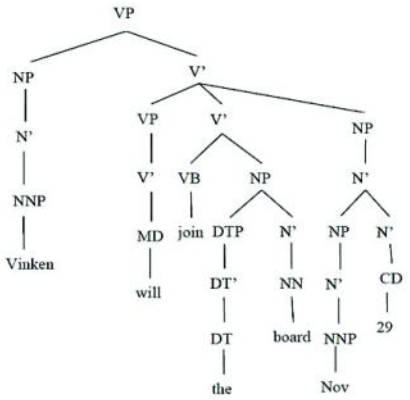
(کاوینگتن، ۱۹۹۴) و (کالینز و همکاران، ۱۹۹۹) از سه قانون ساده استفاده کرده و با کمک نظریه ایکس تیره درخت وابستگی را به درخت سازه‌ای معادل تبدیل

[1] Free word order

می‌کنند. در الگوریتم (کالینز و همکاران، ۱۹۹۹) تمام وابسته‌های یک هسته تنها یک بار منشعب می‌شوند و درنتیجه وابسته‌های یک هسته در یک سطح قرار می‌گیرند، که سبب می‌شود درختی به‌طورتقریبی مسطح به دست آید. شکل (۴) درخت سازه‌ای حاصل از تبدیل درخت وابستگی شکل (۳) را با کمک راهکار (کالینز و همکاران، ۱۹۹۹) نشان می‌دهد. راه‌کار (کاوینگتن، ۱۹۹۴) بر خلاف راه‌کار (کالینز و همکاران، ۱۹۹۹) به‌ازای هر وابسته چندین انشعاب انجام می‌دهد. شکل (۵) درخت سازه‌ای حاصل از اجرای راه‌کار (کاوینگتن، ۱۹۹۴) را بر روی درخت وابستگی شکل (۳) نشان می‌دهد. روش (کالینز و همکاران، ۱۹۹۹) با محدودکردن تعداد انشعاب‌ها به حداقل (یعنی یک انشعاب به‌ازای هر رابطه وابستگی) سبب کاهش فراخوانی شده است. روش (کاوینگتن، ۱۹۹۴) با انجام حداکثر انشعاب سبب کاهش دقت شده است.



(شکل- ۴): درخت سازه‌ای حاصل از الگوریتم تبدیل (کالینز و همکاران، ۱۹۹۹)



(شکل- ۵): درخت سازه‌ای حاصل از الگوریتم تبدیل (کاوینگتن، ۱۹۹۴)

در الگوریتم پیشنهادی (ژیا و پالمر، ۲۰۰۱) حداقل انشعاب و حداکثر انشعاب برای همهٔ وابسته‌ها پذیرفته نیست. این الگوریتم با اضافه‌کردن سه جدول و با کمک نظریهٔ ایکس تیره، به‌ازای هر وابسته تعداد معینی انشعاب در نظر گرفته است. اطلاعات مربوط به جداول به‌صورت دستی استخراج شده و الگوریتم بیشتر ساختاری قاعده‌مند دارد. در این الگوریتم از سه جدول گزاره[1]، تغییرات[2] و انشعاب استفاده شده است. جدول گزاره شامل انواع گزاره‌هایی است که یک هسته می‌تواند بگیرد؛ جدول تغییرات فهرست انواع تغییراتی است که هسته می‌تواند داشته باشد. جدول انشعاب مشخص می‌کند که هر برچسب نحوی به چه برچسب نحوی دیگری می‌تواند منشعب شود.

هر سه الگوریتم گفته‌شده فرض کرده‌اند که هر برچسب کلمه تعداد مشخصی انشعاب دارد (حداقل تعداد انشعاب، حداکثر تعداد انشعاب و تعداد مشخص انشعاب براساس جداول). الگوریتم‌ها به‌صورت به‌طورکامل قاعده‌مند تبدیل را انجام می‌دهند و از دیگر اطلاعات موجود در ساختار وابستگی استفاده نمی‌کنند. تعداد مشخص انشعاب و عدم استفاده از اطلاعات موجود در ساختار وابستگی را می‌توان به‌عنوان دو محدودیت موجود در این الگوریتم‌ها در نظر گرفت.

درخت‌بانک با چند نوع نمایش در (پالمر و همکاران، ۲۰۰۹) ارائه شده است. بر پایهٔ این مقاله، درخت‌بانک جدیدی با نام پروبانک[3] ارائه شده است. آن‌ها تلاش کرده‌اند که این درخت‌بانک دارای نسخهٔ سازگار[4] وابستگی و سازه‌ای باشد. بدین منظور برای رفع فاصله بین این دو ساختار به‌همراه ذخیرهٔ این دو ساختار در یک درخت‌بانک، اطلاعات اضافهٔ دیگری نیز ذخیره شده است. در این مقاله ادعا شده است که وجود این اطلاعات برای سازگاربودن نیز مفید است. هر چند که برخی از ناسازگاری‌ها نیاز به اصلاح دستی دارد و نمی‌توان این درخت بانک را به‌طور تمام قاعده‌مند ایجاد کرد.

الگوریتمی نیمه‌خودکار به‌منظور تولید درخت بانک فارسی که تجزیه متون را بر اساس نظریهٔ کمینه‌گرایی چامسکی (چامسکی، ۱۹۹۸) امکان‌پذیر می‌سازد، در (شریفی آتشگاه، ۱۳۸۸) توضیح داده شده است. الگوریتم معرفی‌شده به‌صورت پایین به بالا بوده و در ارزیابی آن تنها به گروه اسمی اکتفا شده است.

(کلارک و کوران، ۲۰۰۹) به مقایسهٔ بهترین تجزیه‌گر CCG[5] زمان خود با تجزیه‌گر CCG بر پایهٔ

[1] Argument
[2] Modification
[3] ProbBank
[4] Consistence
[5] Combinatory Categorial Grammar

درخت‌بانک پن پرداختند. آن‌ها برخی مشکلات این تجزیه‌گر را بررسی کرده و راه‌کاری در جهت بهبود عملکرد تجزیه‌گر ارائه کردند. (کامرفلد و همکاران، ۲۰۱۲) با کمک راه‌کار ارائه‌شده در (کلارک و کوران، ۲۰۰۹) توابع و راه‌کاری جدید تعریف کردند، که به‌کمک آن به ادغام مقوله‌های[1] مورد استفاده در تبدیل CCG به سازه‌ای پرداختند و توانستند کیفیت تبدیل را بهبود ببخشند.

(کیو و همکاران، ۲۰۱۴) برای زبان چینی، یک درخت بانک با چند نوع نمایش ارائه داده‌اند. مهم‌ترین نمایش‌ها شامل ساختار وابستگی و سازه‌ای است. در این مقاله با کمک ساختارهای موجود، تجزیه‌گر نحوی برای زبان چینی آموزش و نشان داده شده است که نتایج به‌دست‌آمدهٔ حاصل از ارزیابی تجزیه‌گر، به‌خوبی تجزیه‌گر استنفورد برای زبان چینی است.

در (سلطان‌زاده و دیگران، ۱۳۹۳) روشی برای تبدیل درخت وابستگی به درخت سازه‌ای معادل آن در فارسی به کمک یک الگوریتم قاعده‌مند ارائه شده است. به‌منظور آزمایش الگوریتم ارائه شده یکصد جمله به‌صورت تصادفی از درخت‌بانک Perdt انتخاب شده و به‌صورت دستی به درخت سازه‌ای تبدیل شده است. با کمک یکصد جملهٔ سازه‌ای، روش ارائه شده مورد ارزیابی قرار گرفته است و میزان کیفیت نهایی برابر با ۹۶/۰۵ گزارش شده است. روش بیان‌شده در این مقاله با توجه به قواعد تعریف‌شده مختص زبان فارسی است و نمی‌توان در دیگر زبان‌ها مورد استفاده قرار داد. در این مقاله بیان شده است که راه‌کار ارائه شده قادر به تبدیل برخی از ساختارها نظیر منادا، بند وصفی و... نیست.

(بات و همکاران، ۲۰۱۲) الگوریتمی جهت استخراج دستور درخت الحاقی[2] از ساختار وابستگی و ساختار سازه‌ای معادلش، با کمک راه‌کاری مشابه راه‌کار بیان‌شده در (ژیا و همکاران، ۲۰۰۹)، ارائه کرده‌اند. الگوریتم ارائه‌شده در این مقاله با استفاده از قوانین تبدیل، ساختار وابستگی را به دستور درخت الحاقی معادل تبدیل می‌کند.

الگوریتمی بر پایهٔ (ژیا و همکاران، ۲۰۰۹) در (بات و ژیا، ۲۰۱۲) ارائه شده است، که به ارائهٔ ساختار وابستگی با اطلاعات بیشتر نسبت به ساختار وابستگی ساده پرداخته و آن را DS plus نامیده‌اند. بر اساس این مقاله برای تبدیل با کیفیت بالاتر ساختار وابستگی به ساختار سازه‌ای معادل، باید برخی اطلاعات اضافه ذخیره شود که این اطلاعات به همراه ساختار وابستگی در درخت بانک پروبیانک ذخیره شده است.

راه‌کار ارائه‌شده در (ژیا و همکاران، ۲۰۰۹) به‌عنوان یک راه‌کار ترکیبی قاعده‌مند و آماری به‌شمار می‌آید که به دو مرحلهٔ آموزش و ارزیابی تقسیم می‌شود. این الگوریتم بر روی زبان انگلیسی مورد آزمایش و ارزیابی قرار گرفته است. الگوریتم ارائه‌شده به‌دلیل عدم استفاده از ویژگی‌های خاص هر زبان قابلیت اجرا بر روی زبان‌های غیر انگلیسی را نیز دارد.

۳- تبدیل خودکار ساختار

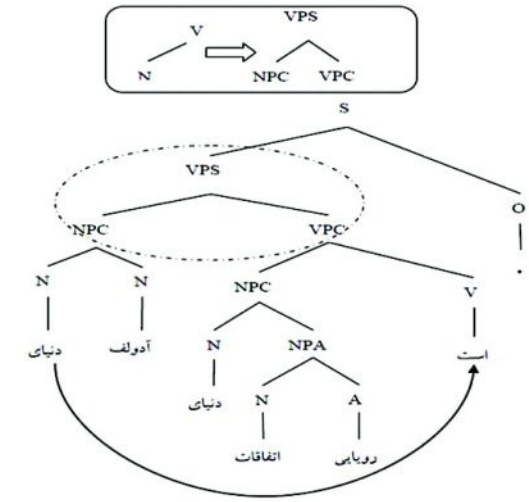
الگوریتم ارائه‌شده در (ژیا و همکاران، ۲۰۰۹) ترکیبی از روش‌های آماری و قاعده‌مند است. راه‌کار ارائه‌شده در این مقاله دارای دو مرحله است.

در مرحلهٔ نخست به آموزش مدل تبدیل پرداخته و اطلاعات آماری جمع‌آوری می‌شود. در این مرحله از الگوریتم، درخت وابستگی و درخت سازه‌ای معادل به‌عنوان ورودی دریافت می‌شود و به‌ازای هر یال وابستگی موجود در درخت وابستگی یک قانون تبدیل تشکیل می‌شود. قانون تبدیل، درخت دوتایی شامل سه گره است، که با توجه به موقعیت دو کلمه دو سر یال وابستگی، در درخت سازه‌ای انتخاب می‌شود. شکل (۶) قانون تبدیل به‌دست‌آمده برای وابسته "دنیای" (با برچسب کلمه N) و هسته "است" (با برچسب کلمه V) را نشان می‌دهد[3]. برای یافتن قانون تبدیل اولین پدر مشترک دو کلمه به همراه فرزند چپ و راست پدر مشترک انتخاب می‌شود. پس از پایان این مرحله مجموعه‌ای از یال‌ها، به همراه قوانین تبدیل و فراوانی آن‌ها را خواهیم داشت.

در مرحلهٔ دوم، الگوریتم درخت وابستگی را به‌عنوان ورودی دریافت می‌کند و سپس با کمک قوانین تبدیل به‌دست‌آمده در مرحلهٔ نخست شروع به تبدیل درخت وابستگی به درخت سازه‌ای معادل آن می‌کند.

[1] Category
[2] Tree-adjoining grammar

[3] به‌دلیل استفاده از برخی ابزارهای پردازش زبان، مانند تجزیه‌گر سازه‌ای که نوشتار ورودی را به‌صورت چپ به راست دریافت می‌کنند، و الگوریتم مورد استفاده در این مقاله، بدون از دست رفتن هیچ‌گونه اطلاعاتی نمایش ترتیب کلمات در جملات فارسی نیز از چپ به راست است.



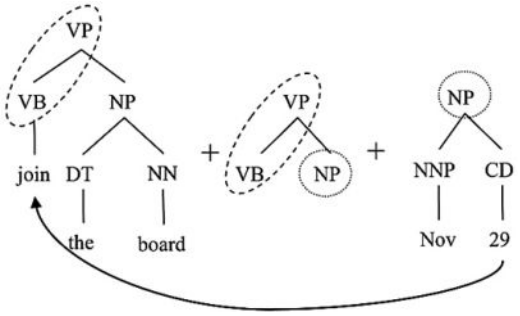
(شکل- ۶): به‌دست‌آوردن قانون تبدیل (درون کادر مستطیل) برای وابسته "دنیای" و هسته "است"

شبه‌برنامهٔ مربوط به مرحلهٔ دوم الگوریتم به‌صورت زیر است:

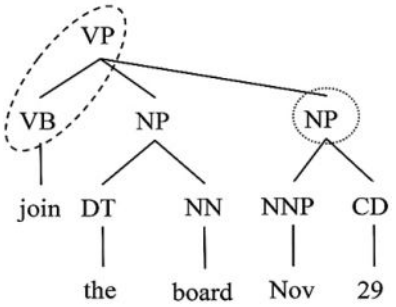
۱- اگر گره X برگ بود
۲- درخت سازه‌ای X با تنها یک گره را برگردان
۳- برای هر گره Y از فرزندان X
۴- درخت سازه‌ای T_Y را بساز
۵- برای هر فرزند چپ (Y) از گره X به‌ترتیب از راست به چپ
۶- قانون تبدیل مربوط به یال (X,Y) را انتخاب کن
۷- درخت T_Y را با کمک قانون تبدیل به درخت کنونی T_X متصل کن
۸- مرحله ۵ تا ۷ را برای هر فرزند راست (Y) از گره X به ترتیب از چپ به راست اجرا کن.

الگوریتم ابتدا از هستهٔ درخت وابستگی شروع به کار می‌کند؛ سپس به‌سراغ فرزندان چپ هسته رفته از راست به چپ این فرزندان پیمایش و در ادامه به‌سراغ فرزندان راست رفته و از چپ به راست، پیمایش می‌شوند. درصورتی‌که هریک از فرزندان دارای وابسته باشند، با آن‌ها مانند یک درخت وابستگی مجزا رفتار و الگوریتم برای فرزندان وابسته فراخوانی می‌شود. خروجی الگوریتم، که یک درخت سازه‌ای است، با قانون تبدیل انتخاب‌شده برای یال وابستگی و درخت مربوط به هسته، ادغام می‌شود. مکان اتصال زیردرخت بازگشتی به درخت اصلی با توجه به مکان قرار گیری هستهٔ درخت وابستگی مشخص می‌شود. ادغام دو درخت سازه‌ای به معنای آن است که دو درخت در مکان اتصال به‌واسطهٔ قانون تبدیل به هم متصل می‌شوند.

درصورتی‌که دو درخت در محل اتصال دارای یال یا گرهٔ مشترک با قانون تبدیل باشند. یال‌ها (گره‌ها) با یکدیگر ادغام می‌شوند. به‌عنوان مثال شکل (۷) اتصال زیردرخت بازگشتی مربوط به وابسته "29" به درخت سازه‌ای در حال شکل‌گیری هسته، "join" را در مکان اتصال گرهٔ VP نشان می‌دهد. درخت نهایی به‌صورت شکل (۸) تشکیل می‌شود. در این درخت یال و گرهٔ مشخص‌شده ادغام می‌شوند.



(شکل- ۷): ادغام قانون تبدیل (درخت وسط) با درخت‌های هسته (درخت چپ) و وابسته (درخت راست)



(شکل- ۸): درخت سازه‌ای بعد از ادغام قانون تبدیل با درخت‌های وابسته و هسته در شکل (۷)

ساخت مجموعه قوانین تبدیل با استفاده از مجموعه درخت‌های وابستگی و سازه‌ای معادل، انجام می‌شود. به همین دلیل امکان دارد در هنگام ساخت قوانین تبدیل برای یک یال وابستگی از یک درخت، یک قانون تبدیل و از درخت دیگر، قانون تبدیل دیگری به‌دست آید. درنهایت ممکن است بیش از یک قانون تبدیل برای آن یال وابستگی به‌دست آید، که باعث ایجاد ابهام در مرحلهٔ دوم خواهد شد. برای رفع این مشکل، اطلاعات اضافی موجود در ساختار وابستگی، مانند نوع وابستگی، به‌همراه قانون تبدیل، ذخیره می‌شود. بنابراین درهنگامی‌که الگوریتم نیاز به یافتن قانون تبدیل برای یک یال وابستگی دارد و برای آن یال بیش از

یک قانون تبدیل وجود داشته باشد، برای ایجاد تمایز میان قوانین، از اطلاعات زیر به‌ترتیب استفاده می‌شود:

۱- آیا قانون تبدیل مربوط به یال وابستگی فرزند چپ هسته است یا راست؟

۲- نوع وابستگی مربوط به رابطهٔ وابستگی آن قانون چه بوده است؟

۳- آیا وابسته، خود دارای وابسته (وابستهٔ وابسته به هسته) است؟

۴- آیا بین وابسته و هسته، وابستهٔ دیگری وجود دارد؟

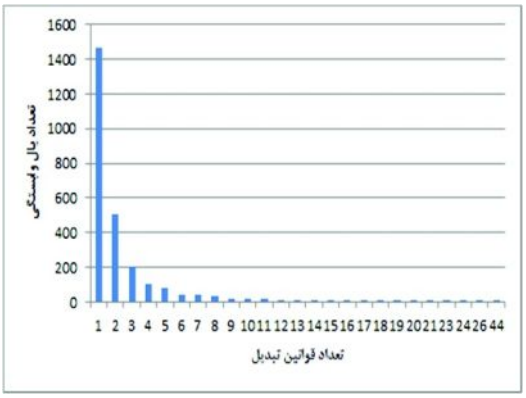
پس از بررسی اطلاعات ذکرشده درصورتی‌که هنوز ابهام وجود داشته باشد، پربسامدترین قانون تبدیل انتخاب می‌شود. نمودار نمایش داده‌شده در شکل (۹) تعداد قوانین تبدیل برحسب فراوانی تعداد یال وابستگی را برای یک نمونه از مجموعه قوانین تبدیل فارسی بدون استفاده از ویژگی‌های ذکرشده نشان می‌دهد. محور افقی، فراوانی قوانین تبدیل و محور عمودی تعداد یال وابستگی را نشان می‌دهد. به‌عنوان مثال عدد سه در محور افقی به این معناست که برای ۲۰۶ یال وابستگی سه نوع متفاوت قانون تبدیل داریم. شکل (۱۰) نمودار تعداد قوانین تبدیل برحسب فراوانی تعداد یال وابستگی را با درنظرگرفتن ویژگی‌های ذکرشده، نشان می‌دهد. همان‌گونه که مشاهده می‌شود در این حالت تمایز میان یال‌های وابستگی افزایش یافته و تعداد یال‌های وابستگی، تنها با یک قانون تبدیل افزایش یافته است.

این الگوریتم دارای برخی مشکلات برای تبدیل ساختار وابستگی به سازه‌ای است. برای هر دو زبان فارسی و انگلیسی یکصد جمله به‌صورت تصادفی از درخت بانک وابستگی فارسی (قیومی، ۲۰۱۲a) و درخت بانک انگلیسی پن (مارکوس و همکاران، ۱۹۹۳) انتخاب می‌کنیم. سپس ساختار وابستگی این جملات را به ساختار سازه‌ای معادل تبدیل کرده و با درخت اصلی موجود در درخت بانک مقایسه می‌کنیم. میانگین طول جملات انتخاب‌شده، در فارسی ۲۴ کلمه و در انگلیسی ۱۸ کلمه است. پس از بررسی خطاهای یافت‌شده انواع خطاها را در سه دستهٔ اصلی تقسیم‌بندی می‌کنیم:

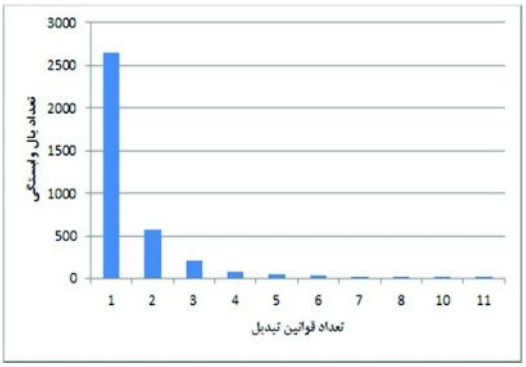
۱- خطاهای حاصل از انتخاب غلط قانون تبدیل یا عدم دیده‌شدن قانون تبدیل که از مجموع کل خطاها، ۴۹/۴ درصد در فارسی و ۵۰/۸ درصد در انگلیسی مربوط به این نوع خطا می‌شود.

۲- خطاهای حاصل از مکان اتصال اشتباه که میزان آن در فارسی ۳۵ درصد و در انگلیسی ۳۷/۴ درصد است.

۳- خطاهای حاصل از ادغام غلط زیر درختان با یکدیگر، با فراوانی ۱۵/۶درصد در فارسی و ۱۱/۸ درصد در انگلیسی.



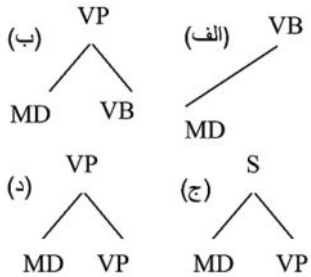
(شکل- ۹): نمودار تعداد یال وابستگی برحسب تعداد قوانین تبدیل بدون درنظرگرفتن ویژگی‌ها



(شکل- ۱۰): نمودار تعداد یال وابستگی برحسب تعداد قوانین تبدیل بدون درنظرگرفتن ویژگی‌ها

حتی پس از بررسی اطلاعات ذخیره‌شده در کنار قانون تبدیل و انتخاب قانون با تکرار بیشتر، ممکن است که قانون انتخاب‌شده برای یال وابستگی در آن جمله اشتباه باشد؛ که همان خطای دستهٔ نخست است. همچنین امکان دارد برخی از یال‌های وابستگی در مجموعه قوانین تبدیل، وجود نداشته باشد که می‌تواند به‌دلیل عدم مشاهده یال وابستگی در داده آموزش یا به‌عبارتی همان حجم کوچک دادهٔ آموزش باشد. مجموعهٔ این عوامل سبب ایجاد خطا در درخت سازه‌ای نهایی خواهد شد. به‌عنوان مثال، برای یال

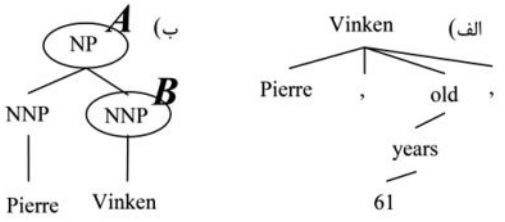
وابستگی، برای وابستهٔ will و هستهٔ join در جملهٔ مثال زده‌شده در شکل (۳)، در مجموعه قوانین تبدیل، سه قانون تبدیل به‌دست می‌آید که در شکل (۱۱) نمایش داده شده است. همان‌طورکه گفته شد در این حالت، اطلاعات دیگر که در هنگام به‌دست‌آوردن قوانین تبدیل ذخیره شده، مورد بررسی قرار می‌گیرد. از میان سه قانون تبدیل، دو قانون تبدیل (۱۱-ب) و (۱۱-د) در اطلاعات ذکرشده یکسان هستند. اکنون پربسامدترین قانون انتخاب می‌شود و چون این انتخاب برای همهٔ درخت‌ها ثابت است، امکان انتخاب قانون با تکرار کمتر و با اطلاعات یکسان وجود نخواهد داشت.



(شکل- ۱۱): (الف) یال وابستگی انتخاب‌شده از شکل (۳)، (ب-د) سه قانون تبدیل موجود در مجموعه قوانین برای یال وابستگی شکل (۳-الف)

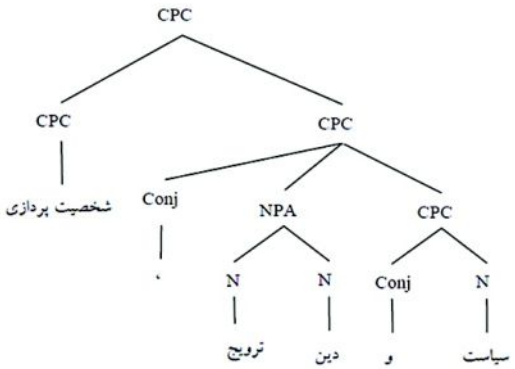
برای حل مشکل خطای نوع نخست باید سعی کرد تمایز میان قوانین تبدیل افزایش پیدا کند. گرچه این امر به صورت صددرصدی بعید به‌نظر می‌آید؛ ولی می‌توان با اضافه‌کردن اطلاعات دیگر شانس انتخاب قوانین با تکرار کمتر را نیز به وجود آورد. بررسی این مشکل و ارائهٔ راه‌کاری در جهت رفع آن می‌تواند به‌عنوان کارهای آتی در نظر گرفته شود.

دستهٔ دوم که حاصل از اشتباه در مکان اتصال است، به‌خاطر ثابت‌بودن ترتیب اتصال فرزندان و محل اتصال زیردرخت‌های مربوط به هر مرحله است. مطابق با روش استفاده‌شده، ترتیب ملاقات وابسته‌های یک هسته ابتدا از وابسته‌های چپ و سپس وابسته‌های راست است. همچنین مکان اتصال برای هر زیردرخت ثابت فرض می‌شود. به‌عنوان مثال در شکل (۱۲) می‌خواهیم وابسته‌های ","، "old" و "," را به زیردرخت هسته "Vinken" متصل کنیم. مطابق با الگوریتم محل اتصال با هسته در نقطه B خواهد بود، درحالی‌که این مکان می‌تواند هریک از نقاط A و B باشد و در مثال ذکرشده مکان اتصال صحیح نقطهٔ A است. فرض ثابت‌بودن این نقطه باعث ایجاد خطا می‌شود.

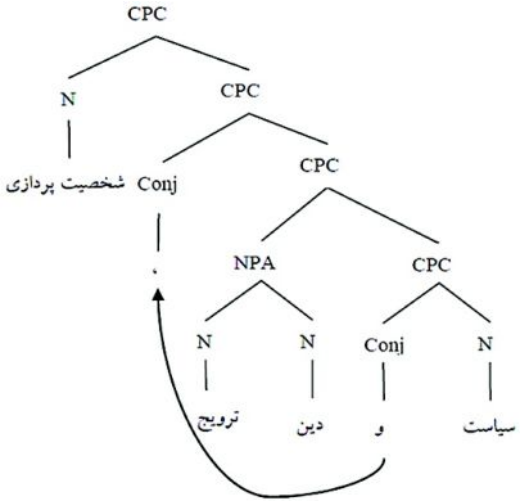


(شکل- ۱۲): (الف) قسمتی از ساختار وابستگی مربوط به یک جمله (ب) درخت سازه‌ای مربوط به هسته و فرزند چپ(Pierre) ساختار وابستگی (الف)

خطاهای حاصل از ادغام غلط، مربوط به ساختار جملات می‌شود. به‌عنوان مثال ممکن است در ساختار سازه‌ای، دو گره با وجود آن‌که دارای برچسب یکسان بوده، ولی ادغام نشده باشند. درصورتی‌که با توجه به الگوریتم، ادغام صورت می‌گیرد. در عبارت "شخصیت‌پردازی، ترویج دین و سیاست" در هنگام اتصال وابستهٔ "و" به هسته "،" درخت حاصل از اتصال به شکل (۱۳) در خواهد آمد. درحالی‌که ساختار سازه‌ای به‌صورت شکل (۱۴) است. ادغام در این حالت به‌صورت قاعده‌مند صورت می‌گیرد؛ درحالی‌که در برخی موارد، همانند مثال ذکرشده نیاز به ادغام نیست و ادغام سبب ایجاد خطا می‌شود. می‌توان به‌عنوان کارهای مورد بررسی در آینده، با بررسی ویژگی‌های ساختاری این جملات، در هنگام اجرای الگوریتم در مورد ادغام یا عدم ادغام تصمیم‌گیری کرد.



(شکل- ۱۳): ساختار سازه‌ای به‌دست‌آمده از الگوریتم تبدیل بعد از ادغام



(شکل- ۱۴): ساختار سازه‌ای موجود در درخت بانک PerTreebank

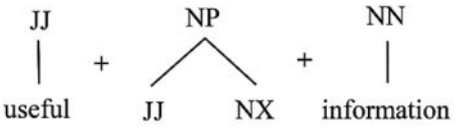
گره‌های موجود در درخت‌بانک سازه‌ای ممکن است، دارای صفر (گره برگ)، یک، دو یا بیشتر فرزند باشند. تشکیل این گره‌ها در درخت سازه‌ای حاصل از تبدیل الگوریتم با کمک عمل ادغام صورت می‌گیرد. در جدول (۲) درصد فراوانی انواع انشعاب[1] موجود در درخت‌بانک فارسی و انگلیسی برای نمونه‌ای تصادفی هزار جمله‌ای نمایش داده شده است. مطابق با جدول (۲) در انگلیسی ۱۹/۱ درصد و در فارسی ۸/۵ درصد گره‌ها تک‌فرزندی هستند. دستهٔ نخست و دوم خطاها منجر به نوع خاصی از خطا می‌شود که مرتبط با گره‌های تک‌فرزندی هستند. اگر محل اتصال غلط انتخاب شود، یا قانون تبدیل به‌درستی انتخاب نشود، ممکن است سبب شود که ادغام انجام نشود (یا ادغام به غلط صورت گیرد). درحالتی‌که ادغام لازم بوده ولی به‌دلیل خطای نوع نخست یا دوم این ادغام صورت نگیرد، ممکن است منجر به ایجاد یک گره با تنها یک یال شود، که گرهٔ سازه‌ای تکی است. درنتیجه گره‌های سازه‌ای تکی به‌وجود می‌آید که در درخت سازه‌ای نباید وجود داشته باشد.

(جدول- ۲): درصد فراوانی انواع انشعاب برای گره‌های داخلی در فارسی و انگلیسی

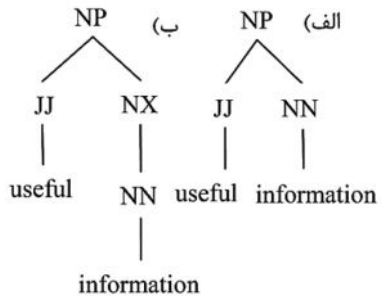
انشعاب	انگلیسی	فارسی
تکی	۱۹/۱	۸/۵
دوتایی	۵۴/۶	۹۰/۵
سه‌تایی و بیشتر	۲۶/۳	۱

[1] Projection type

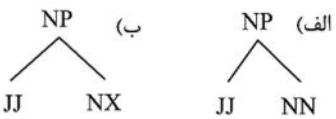
به‌عنوان مثالی از ایجاد گرهٔ سازه‌ای تکی حاصل از خطای نوع نخست، از عبارت "useful information" استفاده می‌کنیم. در این عبارت کلمهٔ useful، وابسته برای هسته، یعنی کلمه information است. در هنگام ساخت درخت سازه‌ای این عبارت توسط الگوریتم توضیح داده‌شده در این بخش شکل (۱۵)، درخت سازه‌ای شکل (۱۶-ب) به‌دست می‌آید. درحالی‌که درخت سازه‌ای معادل ساختار وابستگی، موجود در درخت‌بانک پن به‌صورت شکل (۱۶-الف) است. علت خطای به‌وجود آمده، انتخاب قانون تبدیل غلط است. در درخت شکل (۱۶-ب)، به‌جای انتخاب قانون تبدیل شکل (۱۷-الف)، قانون تبدیل شکل (۱۷-ب) انتخاب شده است. انتخاب غلط قانون تبدیل سبب ایجاد گره تک‌یالی (NX,NN) در درخت سازه‌ای شکل (۱۶-ب) شده است.



(شکل- ۱۵): ادغام قانون تبدیل اشتباه با درخت هسته و وابسته



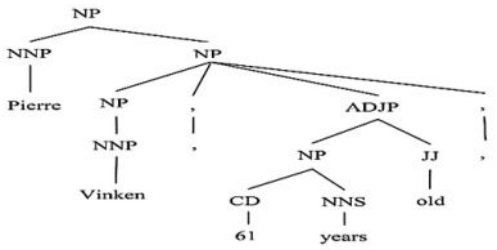
(شکل- ۱۶): الف) قسمتی از درخت سازه‌ای موجود در درخت بانک پن، ب)درخت سازه‌ای حاصل از قانون تبدیل غلط شکل (۱۵)



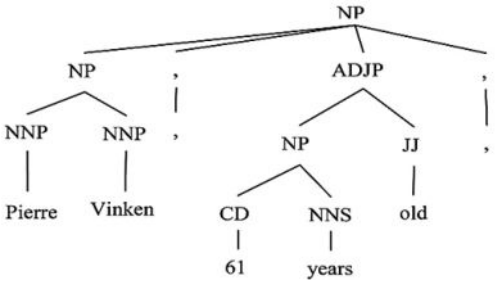
(شکل- ۱۷): دو نمونه قانون تبدیل برای وابسته JJ و هسته NN

خطای نوع دوم نیز می‌تواند عملکردی مشابه خطای نوع نخست در ایجاد گرهٔ سازه‌ای تکی داشته باشد. درصورتی‌که درخت سازه‌ای حاصل از مکان اشتباه شکل (۱۲) رسم شود، شکل (۱۸) به‌دست می‌آید. به‌دلیل محل

اتصال اشتباه انتخاب شده، گرهٔ سازه‌ای تکی به‌وجود آمده است که در ساختار سازه‌ای شکل (۱۹) وجود ندارد.



(شکل- ۱۸): ساختار سازه‌ای خروجی الگوریتم با انتخاب نقطه B در شکل (۱۲-ب)



(شکل- ۱۹): ساختار سازه‌ای موجود در درخت بانک پن

در این مقاله ما برای حل مشکل گره‌های سازه‌ای تکی اقدام به پس‌پردازش بر روی درخت خروجی می‌کنیم. گره‌های سازه‌ای تکی، حاصل از خطاهای نوع نخست و دوم الگوریتم تبدیل هستند. با درنظرگرفتن یک مکاشفه[1] سعی می‌کنیم، گره‌های سازه‌ای تکی موجود در درخت سازه‌ای حاصل از تبدیل را درست‌یابی کرده و درصورت وجود اشتباه، آن‌ها را حذف کنیم. برای این‌کار ابتدا درخت‌بانک مورد استفاده در مرحلهٔ آموزش (مرحلهٔ نخست) را انتخاب می‌کنیم. با پیمایش تمام گره‌های موجود در هریک از درخت‌های سازه‌ای درون درخت بانک، فهرستی از گره‌های تکی به‌دست می‌آید که همراه با برچسب اجزای سخن دو سر یال در فهرستی با نام فهرست گره‌های تکی ذخیره می‌کنیم. با کمک فهرست گره‌های تکی، می‌توانیم با دقت بالایی انشعاب‌های تکی را که از نظر ساختار سازه‌ای صحیح هستند از انشعاب‌های ناصحیح موجود تشخیص دهیم. بدین شکل که در صورت عدم وجود برچسب دو سر یال در فهرست، آن انشعاب قابل انجام نیست و باید حذف شود.

در مرحلهٔ پس‌پردازش، بر روی هریک از درختان، راه‌کار مکاشفه را اجرا کرده و گره‌های سازه‌ای تکی را که در فهرست گره‌های تکی موجود نباشند، حذف می‌کنیم. حذف گرهٔ سازه‌ای تکی به‌معنای حذف گرهٔ پدر و جایگزینی آن با گرهٔ فرزند است. به‌عنوان مثال پس از به‌دست‌آمدن درخت سازه‌ای شکل (۱۶-ب) با اجرای مکاشفه بر روی آن به‌علت عدم وجود برچسب‌های NX و NN در فهرست، گره NX حذف می‌شود و همان درخت سازه‌ای شکل (۱۶-الف) به‌دست می‌آید.

بررسی گره‌های سازه‌ای تکی، در هنگام انتخاب قوانین برای ساخت درخت نیز امکان پذیر است؛ اما نسبت به انجام پس‌پردازش معایبی دارد. اگر گرهٔ سازه‌ای تکی حاصل از عدم دیده‌شدن قانون تبدیل در مرحلهٔ نخست باشد (خطای نوع نخست)، با اضافه‌کردن محدودیت عدم ایجاد گرهٔ سازه‌ای تکی غلط، نمی‌توان قانون درست را انتخاب کرد و درنهایت گرهٔ سازه‌ای تکی اشتباه به‌وجود می‌آید که باید به‌صورت پس‌پردازش حذف شود. همچنین اگر گرهٔ سازه‌ای تکی حاصل از مکان اشتباه اتصال باشد (خطای نوع دوم)، محدودیت انتخابی سبب اصلاح گرهٔ سازه‌ای تکی نمی‌شود و دوباره نیاز به پس‌پردازش، برای حذف گرهٔ سازه‌ای تکی اشتباه است.

گفتیم که مکان اتصال اشتباه در الگوریتم تبدیل، منجر به خطای دسته دوم می‌شود. فرض ثابت‌بودن مکان اتصال و درنظرنگرفتن فرزندان دیگر باعث این خطا می‌شود. در بررسی درخت‌های سازه‌ای خروجی الگوریتم مشاهده کردیم به‌منظور جلوگیری از این خطا باید فرض ثابت‌بودن مکان اتصال حذف شده و مکان اتصال با توجه به دیگر فرزندان مشخص شود. به‌عنوان مثال در شکل (۱۲) در صورتی‌که کلمهٔ Pierre در ساختار جمله وجود نداشت، نقطهٔ B مکان اتصال صحیح برای فرزندان راست بود؛ درحالی‌که با وجود کلمهٔ Pierre، نقطهٔ B دیگر نقطهٔ صحیح برای اتصال فرزند راست نیست و مکان اتصال صحیح نقطهٔ A است.

روش پیشنهادی ما در این قسمت برای انتخاب صحیح مکان اتصال، استفاده از یک طبقه‌بند است. به‌منظور تعیین نقطهٔ اتصال صحیح ما یک طبقه‌بند را آموزش دادیم که از بین مکان‌های اتصال ممکن، نقطهٔ مناسب را تعیین کند. بدین منظور مکان‌های اتصال را به‌عنوان طبقه در نظر می‌گیریم.

در بررسی‌هایی که بر روی خطاهای دستهٔ دوم انجام داده‌ایم، مشخص شد که تعریف دو مکان اتصال برای حل مشکل مکان اتصال اشتباه، کافی است. مکان اتصال نخست

[1] Heuristic

یا همان طبقه یک، مانند مکان انتخاب‌شده در (ژیا و همکاران، ۲۰۰۹) انتخاب می‌شود. انتخاب این طبقه به این معنا است که بدون درنظرگرفتن فرزندان سمت دیگر، مکان اتصال، همان آخرین محل اتصال وابستهٔ نزدیک‌تر به هسته خواهد بود. دومین طبقه اما، با توجه به فرزندان سمت دیگر مشخص می‌شود. انتخاب طبقه دو به این معنا است که مکان اتصال در بالاترین نقطه از درخت تشکیل‌شده تا آن زمان خواهد بود. در شکل (۱۲) نقطهٔ B در طبقه یک و نقطهٔ A در طبقه دو قرار می‌گیرد. همان‌گونه که می‌توان از این مثال نیز حدس زد، درصورتی‌که وابستهٔ Pierre وجود نداشت، طبقه یک و دو، هر دو به یک نقطه اشاره می‌کردند. در بخش بعد با توجه به نتایج آزمایش‌ها، نشان خواهیم داد که با استفاده از طبقه‌بند و دو مکان اتصال تعریف‌شده، خطای حاصل از مکان اتصال اشتباه، کاهش زیادی خواهد یافت.

به‌منظور استفاده از طبقه‌بند، نیاز به استخراج ویژگی است. برای تعریف ویژگی به‌سراغ دادهٔ آموزش استفاده‌شده در مرحلهٔ نخست الگوریتم می‌رویم و همان داده‌های استفاده‌شده برای استخراج قوانین تبدیل را استفاده می‌کنیم. برای هر یال وابستگی در درخت وابستگی ویژگی‌هایی که برای این طبقه‌بند استخراج می‌شود در جدول (۳) ارائه می‌شود. نتایج حاصل از آزمایش‌ها در بخش ۴ نشان از کارآیی طبقه‌بند پیشنهادی دارد.

(جدول- ۳): ویژگی‌های مورداستفاده برای طبقه‌بند

نام	توضیح ویژگی	دامنه
F1	برچسب اجزای سخن مربوط به هسته	تمام برچسب‌های اجزای کلام
F2	برچسب اجزای سخن مربوط به وابسته	تمام برچسب‌های اجزای کلام
F3	نوع وابستگی	تمام برچسب‌های وابستگی
F4	جهت قرارگیری وابسته نسبت به هسته	۱، ۰
F5	وجود وابسته بین وابسته و هسته	۱، ۰
F6	آیا وابسته در درخت وابستگی برگ است؟	۱، ۰

تا اینجا روش تبدیل ساختار وابستگی به سازه‌ای توضیح داده شد. با انتخاب نمونه‌ای تصادفی، خطاهای حاصل از الگوریتم تبدیل در زبان فارسی و انگلیسی همراه با مثال، نشان داده شد و خطاها در سه دستهٔ اصلی طبقه‌بندی شد؛ سپس برای نوع خاصی از خطاهای حاصل از دستهٔ نخست و دوم راه‌کاری مکاشفه‌ای ارائه شد. همچنین برای دستهٔ دوم از خطاها که حاصل از مکان اتصال اشتباه بود، استفاده از طبقه‌بند پیشنهاد شد. در بخش بعد به بررسی عملکرد الگوریتم تبدیل برای زبان فارسی و انگلیسی به‌همراه راه‌کارهای پیشنهادی در این بخش می‌پردازیم. نشان خواهیم داد استفاده از راه‌کارهای گفته‌شده، موجب بهبود عملکرد الگوریتم در هر دو زبان فارسی و انگلیسی خواهد شد.

۴- ارزیابی

در این بخش ابتدا به بررسی کیفیت طبقه‌بند پرداخته، سپس به توضیح نتایج حاصل از عملکرد الگوریتم تبدیل می‌پردازیم. در آزمایش‌های انجام‌شده، آزمایش بر روی زبان انگلیسی همانند (ژیا و همکاران، ۲۰۰۹) بر بخش ۰ (صفر) درخت‌بانک پن انجام شده است. همچنین برای زبان فارسی از درخت‌بانک‌های PerTreebank و DepPerTreebank که در بخش ۱ معرفی شد به‌عنوان دادهٔ آموزش و ارزیابی استفاده شده است. در هر دو زبان فارسی و انگلیسی از هشتاد درصد درخت‌های موجود در درخت‌بانک به‌عنوان دادهٔ آموزش و از مابقی به‌عنوان دادهٔ ارزیابی استفاده شده است. راه‌کار (ژیا و همکاران، ۲۰۰۹) به‌عنوان روش مبنا در هر دو زبان انگلیسی و فارسی در نظر گرفته شده است.

به‌منظور ارزیابی الگوریتم تبدیل، درخت‌بانک‌ها به دو بخش آموزش و ارزیابی تقسیم شده است. بعد از به‌دست‌آوردن نتایج، برای به‌دست‌آوردن درخت‌بانک با حجم بزرگ‌تر، از همهٔ درخت‌بانک به‌عنوان دادهٔ آموزش استفاده شده و درخت‌بانک Perdt با سی‌هزار جمله به درخت‌بانک سازه‌ای معادل تبدیل شده است. در مرحلهٔ بعد، از همهٔ سی‌هزار جمله درخت بانک تبدیل شده به‌عنوان دادهٔ آموزش و ارزیابی تجزیه‌گر استنفورد (کلاین و منینگ، ۲۰۰۳) استفاده شده است. از سی‌هزار جمله درخت‌بانک تبدیل‌شده توسط الگوریتم تبدیل، ۲۷,۰۰۰ جملهٔ آن به‌عنوان دادهٔ آموزش و سه‌هزار جمله به‌عنوان دادهٔ ارزیابی مورد استفاده قرار گرفت.

از آن‌جایی‌که ویژگی‌های معرفی‌شده جهت استفاده در طبقه‌بند، از دسته ویژگی‌های گسسته محسوب می‌شوند، در این قسمت از طبقه‌بند درخت تصمیم، که

برای این دسته از ویژگی‌ها مناسب است، استفاده می‌کنیم. جدول (۴) کیفیت طبقه‌بند را براساس ویژگی‌ها به‌صورت مجزا و ترکیب همهٔ ویژگی‌ها برای زبان فارسی نشان می‌دهد. به‌منظور ارزیابی کیفیت طبقه‌بند برای زبان انگلیسی نیز، نتایج حاصل از ارزیابی طبقه‌بند در جدول (۵) ارائه شده است. کیفیت نهایی طبقه‌بند معرفی‌شده در بخش ۳ برای زبان فارسی و انگلیسی به‌ترتیب برابر با ۹۹/۱ و ۹۸/۲ درصد است.

(جدول- ۴): میانگین وزن‌دار نتایج به‌دست‌آمدهٔ مربوط به دقت، فراخوانی و معیار ترکیبی f1-score برای طبقه‌بند فارسی

نام	دقت	فراخوانی	F-Measure
F1	۷۱/۰	۷۰/۷	۷۰/۸
F2	۸۳/۳	۸۳/۳	۸۳/۳
F3	۸۲/۱	۸۲/۰	۸۲/۰
F4	۸۸/۳	۸۴/۱	۸۶/۱
F5	۸۳/۶	۷۴/۸	۷۸/۹
F6	۳۱/۶	۵۶/۲	۴۰/۴
همه ویژگی‌ها	۹۹/۱	۹۹/۱	۹۹/۱

(جدول- ۵): میانگین وزن‌دار نتایج به‌دست‌آمدهٔ مربوط به دقت، فراخوانی و معیار ترکیبی f1-score برای طبقه‌بند انگلیسی

نام	دقت	فراخوانی	F-Measure
F1	۷۴/۴	۷۴/۹	۷۴/۶
F2	۹۱/۷	۹۱/۶	۹۱/۶
F3	۸۵/۹	۸۵/۰	۸۵/۴
F4	۸۸/۰	۸۱/۵	۸۴/۶
F5	۴۲/۷	۶۵/۳	۵۱/۶
F6	۴۱/۵	۶۴/۶	۵۰/۴
همه ویژگی‌ها	۹۸/۲	۹۸/۲	۹۸/۲

برای سنجش کیفیت الگوریتم تبدیل در جدول (۶) نتایج حاصل از اجرای الگوریتم برای دو زبان فارسی و انگلیسی نمایش داده شده است. نتیجهٔ به‌دست‌آمده از اجرای الگوریتم (ژیا و همکاران، ۲۰۰۹) بر روی فارسی برابر با ۶۶/۲۱ درصد بر حسب معیار f1-score است. تفاوت در ساختار دو درخت بانک و تفاوت‌هایی نظیر بدون ترتیب کلمه‌بودن زبان فارسی، سبب پایین‌بودن کیفیت روش‌های مورد استفاده بر روی زبان فارسی می‌شود. در جدول (۶) الگوریتم تبدیل بهبودیافته با کمک راه‌کار مکاشفه با نام تبدیل‌گر مکاشفه‌ای درج شده است. تبدیل‌گر طبقه‌بند نیز اشاره به تبدیل‌گر مبنا به‌همراه استفاده از طبقه‌بند جهت تعیین مکان اتصال دارد. تبدیل‌گر مکاشفه‌ای به‌همراه طبقه‌بند نیز اشاره به ترکیب دو راه‌کار بالا دارد. کیفیت تبدیل‌گر مکاشفه‌ای این مقاله در زبان انگلیسی، نسبت به روش مبنا، حدود ۱/۱ درصد افزایش یافته است. این میزان بهبود برای فارسی حدود ۱۶/۲ درصد است. همان‌گونه که در جدول (۶) مشاهده می‌شود، عملکرد راه‌کار مکاشفه موجب بهبود نتیجه در زبان فارسی و انگلیسی شده است. گرچه عملکرد مکاشفه در فارسی بسیار بهتر از انگلیسی بوده است. علت این امر را می‌توان در تفاوت بین ساختار درخت‌بانک‌ها و زبان دانست. به‌عنوان مثال یکی از تفاوت‌های مشهود، فراوانی انواع انشعاب در درخت‌بانک فارسی و انگلیسی است. مطابق با جدول (۲) مشاهده می‌شود که حدود ۹۹ درصد گره‌های موجود در درخت‌بانک فارسی تکی یا دوتایی هستند که از این میزان ۹۰/۵ درصد دوتایی هستند. این درحالی است که در درخت بانک انگلیسی این عدد حدوداً برابر با ۷۳/۷ درصد است، که از این میزان ۱۹/۱ درصد دارای یک فرزند و ۵۴/۶ درصد باقیمانده دوتایی هستند. می‌توان نتیجه گرفت در جمله‌ای با تعداد کلمات مساوی درخت فارسی دارای تعداد بیشتری گره نسبت به درخت انگلیسی با همان تعداد کلمه است. به‌عنوان دلیل دیگر برای کیفیت پایین‌تر در زبان فارسی نسبت به زبان انگلیسی می‌توان به طول جملات اشاره کرد. میانگین طول جملات در درخت‌بانک انگلیسی ۱۸ کلمه و در فارسی ۲۴ است. طولانی‌تربودن طول جمله‌ها سبب افزایش میزان پیچیدگی و سخت‌ترشدن فرآیند تبدیل می‌شود. همان‌گونه که در جدول (۶) می‌توان مشاهده کرد، معیار فراخوانی در تبدیل‌گر مکاشفه‌ای در هر دو زبان فارسی و انگلیسی کاهش پیدا کرده است. علت این امر به‌دلیل کمبود حجم دادگان آموزش و حذف اشتباه گرهٔ سازه‌ای تکی است.

استفاده از طبقه‌بند نیز موجب بهبود کیفیت تبدیل به‌میزان ۳/۲ درصد در انگلیسی و ۱۰/۱ درصد در فارسی شده است. با ترکیب دو راه‌کار مکاشفه‌ای و طبقه‌بند نتیجهٔ نهایی، بهبودی ۲۵/۸۵ و ۴/۳۹ درصدی را به‌ترتیب برای زبان‌های فارسی و انگلیسی نشان می‌دهد.

(جدول- ۶): نتایج حاصل از تبدیل ساختار وابستگی به سازه‌ای در زبان فارسی و انگلیسی

	روش	دقت	فراخوانی	f1-score
انگلیسی	تبدیل‌گر مبنا	۸۹/۱۹	۹۱/۷۶	۹۰/۴۶
	تبدیل‌گر مکاشفه‌ای	۹۱/۸۶	۹۱/۴۱	۹۱/۶۳
	تبدیل‌گر طبقه‌بند	۹۰/۲۹	۹۷/۳۹	۹۳/۷۱
	تبدیل‌گر مکاشفه‌ای به همراه طبقه‌بند	۹۲/۸۵	۹۶/۹۴	۹۴/۸۵
فارسی	تبدیل‌گر مبنا	۵۳/۰۳	۸۸/۱۳	۶۶/۲۱
	تبدیل‌گر مکاشفه‌ای	۸۵/۴۸	۷۹/۶۱	۸۲/۴۴
	تبدیل‌گر طبقه‌بند	۶۴/۸۸	۹۲/۶۵	۷۶/۳۲
	تبدیل‌گر مکاشفه‌ای به همراه طبقه‌بند	۹۵/۳۹	۸۸/۹۵	۹۲/۰۶

در این بخش کارآیی الگوریتم تبدیل مبنا بر روی زبان فارسی نشان داده شد. با توجه به راه‌کارهای پیشنهادی در بخش ۳، برای بهبود عملکرد الگوریتم، نشان دادیم که کارآیی الگوریتم تبدیل با استفاده از روش مکاشفه و طبقه‌بند برای هر دو زبان فارسی و انگلیسی بهبود یافت. در ادامه با کمک الگوریتم تبدیل و راه‌کارهای توضیح داده‌شده در بخش ۳، درخت‌بانک وابستگی Perdt را به درخت‌بانک سازه‌ای معادل تبدیل و به‌عنوان ورودی تجزیه‌گر سازه‌ای احتمالاتی استنفورد استفاده می‌کنیم. به‌دلیل عدم وجود درخت‌بانک سازه‌ای با حجم بزرگ، درخت‌بانک سازه‌ای تولیدشده به این روش را به رایگان در اختیار عموم قرار داده‌ایم[1].

۵- تجزیه‌گر سازه‌ای فارسی

در این مقاله برای ساخت تجزیه‌گر سازه‌ای زبان فارسی، از تجزیه‌گر سازه‌ای احتمالاتی استنفورد استفاده کرده‌ایم (کلاین و منینگ، ۲۰۰۳). این تجزیه‌گر به زبان جاوا و به‌صورت رایگان موجود است. (قیومی، ۲۰۱۲b) و (قیومی، ۲۰۱۴) نیز این تجزیه‌گر را با استفاده از درخت‌بانک PerTreebank آموزش و ارزیابی کرده‌اند.

در بخش ۴ عملکرد الگوریتم تبدیل بر روی زبان فارسی نشان داده شد. ارزیابی الگوریتم تبدیل برای زبان فارسی با استفاده از درخت‌بانک PerTreebank صورت گرفت. اکنون برای ساخت دادهٔ مورد نیاز ورودی، برای آموزش تجزیه‌گر استنفورد، از کل درخت‌بانک PerTreebank به‌عنوان دادهٔ آموزش ورودی الگوریتم استفاده می‌کنیم و کل درخت‌بانک Perdt را به درخت‌بانک سازه‌ای معادل تبدیل می‌کنیم. برای آموزش و ارزیابی تجزیه‌گر از درخت‌بانک سازه‌ای به‌دست‌آمدهٔ حاصل از تبدیل درخت‌بانک Perdt استفاده کرده و به روش ارزیابی حدّ وسط ده‌تایی[2] عمل می‌کنیم. نتایج حاصل از آزمایش عملکرد تجزیه‌گر در جدول (۷) آورده شده است. به‌منظور نمایش میزان اثر حجم داده بر روی کیفیت تجزیه‌گر، نتایج ارزیابی حدّ وسط ده‌تایی با استفاده از تعداد جملات متفاوت، از درخت‌بانک تبدیل‌شده، گزارش شده است. با مشاهدهٔ نتایج می‌توان دریافت با افزایش تعداد جملات، کیفیت تجزیه‌گر نیز بهبود یافته است.

(جدول- ۷): نتایج تجزیه‌گر با روش ارزیابی حدوسط ۱۰تایی با استفاده از درخت بانک به‌دست‌آمده با کمک راه‌کار پیشنهادی در این مقاله

تعداد جملات	f1-score
۱،۰۰۰	۴۸/۸۳
۵،۰۰۰	۶۰/۸
۱۰،۰۰۰	۶۲/۸۲
۲۰،۰۰۰	۶۶/۰۶
۳۰،۰۰۰	۶۸/۰۸

(قیومی، ۲۰۱۲b) و (قیومی، ۲۰۱۴) نیز با استفاده از درخت بانک PerTreebank، به ارزیابی کیفیت تجزیه‌گر استنفورد پرداخته‌اند. (قیومی، ۲۰۱۲b) تجزیه‌گر استنفورد را در سه حالت آموزش داده است. در راه‌کار (قیومی، ۲۰۱۲b) سعی شده است با جایگزینی واحد خوشه[3] به جای واحد کلمه کیفیت تجزیه‌گر را بهبود ببخشد. اما کیفیت تجزیه‌گر درحالتی‌که واحد تجزیه کلمه باشد برابر با ۵۰/۰۵ بر حسب معیار f1-socre است. می‌توان یکی از علل پایین‌بودن کیفیت نتیجهٔ گزارش‌شده را کوچک‌بودن اندازهٔ درخت‌بانک آموزش و پیچیدگی زبان فارسی دانست. بررسی این علت به‌همراه دیگر عوامل تأثیرگذار بر روی کیفیت تجزیه‌گر در (قیومی، ۲۰۱۴) انجام شده است. (قیومی، ۲۰۱۴) با بررسی این عوامل و ارائهٔ راه‌کار برای آن‌ها کیفیت ۵۹/۴۲ را به‌عنوان مبنا گزارش کرده است. کیفیت گزارش‌شده با استفاده از درخت بانک PerTreebank و

[1] http://ece.ut.ac.ir/en/node/940?destination=node%2F940

[2] 10-fold cross validation

[3] Cluster

فرض مشخص‌بودن برچسب کلمات است. به‌منظور سنجش بهتر، میزان کارآیی تجزیه‌گر سازه‌ای آموزش‌داده‌شده در این مقاله، نتایج استفاده‌شده در این دو پژوهش که به‌عنوان مبنا در نظر گرفته‌اند، در جدول (۸) نمایش داده شده است. برای نمایش اثر راه‌کارهای ارائه‌شده در بخش ۳ بر روی کیفیت تجزیه‌گر، درخت بانک Perdt را با استفاده از الگوریتم (ژیا و همکاران، ۲۰۰۹) بدون هیچگونه تغییری در الگوریتم، در جدول (۶) نمایش داده‌ایم. با توجه به آن‌که درخت‌بانک‌های استفاده‌شده در (قیومی، ۲۰۱۲b) و (قیومی، ۲۰۱۴) متفاوت از درخت‌بانک تبدیل‌شده در این مقاله است، نتیجهٔ تجزیه‌گر آموزش داده‌شده توسط الگوریتم (ژیا و همکاران، ۲۰۰۹) را به‌عنوان مبنا قرار می‌دهیم. بر این اساس مشاهده می‌شود که کیفیت تجزیه سازه‌ای جملات به‌میزان حدود ۲۱ درصد نسبت به کیفیت مبنای این مقاله افزایش یافته است.

(جدول- ۸): نتایج تجزیه‌گر با روش ارزیابی حد وسط ۱۰تایی

راه‌کار	f1-score
(قیومی، ۲۰۱۲b)	۵۰/۰۵
(قیومی، ۲۰۱۴)	۵۹/۴۲
مبنا (الگوریتم (ژیا و همکاران، ۲۰۰۹))	۴۶/۶۲
این مقاله	۶۸/۰۸

در این مقاله با هدف کاهش اثر اندازهٔ کوچک درخت‌بانک آموزش، از ۲۷,۰۰۰ جمله در مرحلهٔ آموزش تجزیه‌گر و از سه‌هزار جملهٔ باقی‌مانده جهت ارزیابی آن استفاده می‌کنیم. گرچه کیفیت تبدیل جملات ۱۰۰٪ نیست با این حال توانستیم به کیفیت ۶۸/۰۸ درصد برای تجزیه‌گر سازه‌ای دست یابیم. انتظار داریم با افزایش کیفیت تبدیل ساختار وابستگی به سازه‌ای این میزان نیز دوباره بهبود یابد.

۶- نتیجه‌گیری و کارهای آتی

دو ساختار وابستگی و سازه‌ای، کاربردهای فراوانی در فعالیت‌های مرتبط با پردازش زبان طبیعی دارند. به‌دلیل اهمیت وجود درخت‌بانک‌های معادل و هزینهٔ زمانی و مالی تولید درخت‌بانک به‌صورت دستی، تولید درخت بانک به‌صورت قاعده‌مند موردتوجه قرار گرفته است. در این مقاله با بررسی راه‌کارهای موجود در تبدیل قاعده‌مند و آماری، بهترین راه‌کار موجود در تبدیل تا حال حاضر مورد بررسی قرار گرفت و خطاهای آن دسته‌بندی شد. همچنین این الگوریتم بر روی زبان فارسی و انگلیسی اجرا و نتایج گزارش شده است. برای مشکل مکان اتصال اشتباه، استفاده از طبقه‌بند و برای مشکل گره‌های سازه‌ای تکی راه‌کار مکاشفه‌ای پیشنهاد شد. نتایج حاصل از اجرای راه‌کارهای گفته‌شده در این مقاله، نشان از بهبود عملکرد الگوریتم تبدیل داشت. در پایان به‌دلیل عدم وجود درخت‌بانک سازه‌ای با حجم بزرگ در زبان فارسی، درخت‌بانک وابستگی Perdt با کمک الگوریتم تبدیل به درخت‌بانک سازه‌ای معادل تبدیل شده است. سپس با کمک درخت‌بانک به‌دست‌آمده، تجزیه‌گر استنفورد آموزش و ارزیابی شد.

از ۲۷,۰۰۰ جملهٔ موجود برای آموزش تجزیه‌گر و از سه‌هزار جملهٔ باقی‌مانده برای ارزیابی تجزیه‌گر استفاده شد. نتایج حاصل در مقایسه با تجزیه‌گر آموزش‌داده‌شده با مبنای درنظرگرفته‌شده در این مقاله بهبودی درحدود ۲۱ درصدی را نشان می‌دهد.

با بررسی خطاهای موجود در الگوریتم تبدیل مشاهده شد که راه‌کار موجود دارای سه مشکل اساسی است. به نظر می‌رسد با اضافه‌کردن برخی اطلاعات دیگر همراه با قوانین تبدیل به بهبود عملکرد الگوریتم در هنگام انتخاب قوانین می‌توان کمک کرد. همچنین شاید بتوان با اضافه‌کردن برخی ویژگی‌های زبانی موجود در زبان فارسی، این بهبود را افزایش داد. از دیگر مواردی که می‌توان به‌عنوان کارهای آتی معرفی کرد، اصلاح دستی درخت‌بانک تبدیل‌شده با استفاده از روش معرفی‌شده در این مقاله است.

با توجه به آن‌که برخی ساختارها میان دو درخت بانک Perdt و DepPerTreebank متفاوت با یکدیگر است، بررسی این تفاوت‌ها جهت بهبود نتایج به‌عنوان کارهای آینده توصیه می‌شود. همچنین می‌توان حجم کمی از درخت‌بانک Perdt را به درخت‌بانک سازه‌ای معادل تبدیل، کرد. در این صورت با توجه به کیفیت بالای روش تبدیل انتظار می‌رود نتایج حاصل از تجزیهٔ نحوی جملات به‌مراتب بیشتر از نتایج کنونی شود.

۷- منابع

سلطان‌زاده، فاطمه، محمد بحرانی و محرم اسلامی (۱۳۹۳)، دادگان درخت نحوی شریف: دادگان درخت نحوی ساخت

سازهای زبان فارسی. مجموعه مقالات سومین همایش زبان‌شناسی رایانشی ایران، دانشگاه صنعتی شریف، ۲۸-۲۹ آبان.

شریفی آتشگاه، مسعود (۱۳۸۸)، تولید نیمه‌خودکار درخت بانک گروه‌های نحوی در متون فارسی، پایان‌نامه دکتری، دانشگاه ادبیات و علوم انسانی، دانشگاه تهران.

Bhatt, R., & Xia, F. (2012). Challenges in converting between treebanks: a case study from the hutb. In Proceedings of META-RESEARCH Workshop on Advanced Treebanking, in conjunction with LREC-2012, Istanbul, Turkey.

Bhatt, R., Rambow, O., & Xia, F. (2012). Creating a Tree Adjoining Grammar from a Multilayer Treebank. In Proceedings of the 11th International Workshop on Tree Adjoining Grammars and Related Formalisms (TAG+ 11) (pp. 162-170).

Chomsky, N. (1998). Minimalist inquiries: The framework (No. 15). MIT Working Papers in Linguistics, MIT, Department of Linguistics.

Clark, S., & Curran, J. R. (2009). Comparing the accuracy of CCG and Penn Treebank parsers. In Proceedings of the ACL-IJCNLP 2009 Conference Short Papers (pp. 53-56). Association for Computational Linguistics.

Collins, M., Ramshaw, L., Hajič, J., & Tillmann, C. (1999). A statistical parser for Czech. In Proceedings of the 37th annual meeting of the Association for Computational Linguistics on Computational Linguistics (pp. 505-512). Association for Computational Linguistics.

Covington, M. A. (1994). An empirically motivated reinterpretation of Dependency Grammar. arXiv preprint cmp-lg/9404004.

Ghayoomi, M. (2012a). Bootstrapping the Development of an HPSG-based Treebank for Persian. Linguistic Issues in Language Technology, 7(1).

Ghayoomi, M. (2012b). Word clustering for Persian statistical parsing. In Advances in Natural Language Processing (pp. 126-137). Springer Berlin Heidelberg.

Ghayoomi, M. (2014). From HPSG-based Persian Treebanking to Parsing (Doctoral dissertation, Freie Universität Berlin, Germany).

Ghayoomi, M., & Kuhn, J. (2014). Converting an HPSG-based Treebank into its Parallel Dependency-based Treebank. In LREC (pp. 802-809).

Goyal, P., & Kulkarni, A. (2014). Converting Phrase Structures to Dependency Structures in Sanskrit.

Hajic, J. (1998). Building a syntactically annotated corpus: The prague dependency treebank. Issues of valency and meaning, 106-132.

Klein, D., & Manning, C. D. (2003). Accurate unlexicalized parsing. In Proceedings of the 41st Annual Meeting on Association for Computational Linguistics-Volume 1 (pp. 423-430). Association for Computational Linguistics.

Kumar, E. (2011). Natural language processing. IK International Pvt Ltd.

Kummerfeld, J. K., Klein, D., & Curran, J. R. (2012). Robust conversion of ccg derivations to phrase structure trees. In Proceedings of the 50th Annual Meeting of the Association for Computational Linguistics: Short Papers-Volume 2 (pp. 105-109). Association for Computational Linguistics.

Marcus, M. P., Marcinkiewicz, M. A., & Santorini, B. (1993). Building a large annotated corpus of English: The Penn Treebank. Computational linguistics, 19(2), 313-330.

McDonald, R. T., & Pereira, F. C. (2006). Online Lear-ning of Approximate Dependency Parsing Algo-rithms. In EACL.

Oepen, S., Flickinger, D., Toutanova, K., & Manning, C. D. (2004). LinGO redwoods: a rich and dynamic tree- bank for HPSG. In Proceedings of the Workshop on Parseval and Beyond and the 3rd International Conference on Language Resources and Evaluation (LREC'02). Las Palmas, Spain.

Palmer, M., Bhatt, R., Narasimhan, B., Rambow, O., Sharma, D. M., & Xia, F. (2009). Hindi syntax: Annotating dependency, lexical predicate-argument structure, and phrase structure. In The 7th International Conference on Natural Language Processing (pp. 14-17).

Pollard, C., & Sag, I. A. (1994). Head-driven phrase structure grammar. University of Chicago Press.

Qiu, L., Zhang, Y., Jin, P., & Wang, H. (2014). "Multi-view Chinese Treebanking." 257–268.4

Rasooli, M. S., Kouhestani, M., & Moloodi, A. (2013). Development of a Persian syntactic dependency treebank. In Proceedings of the 2013 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies (pp. 306-314).

Xia, F. (2001). Automatic grammar generation from two different perspectives (Doctoral dissertation, University of pennsylvania).

Xia, F., & Palmer, M. (2001). Converting dependency structures to phrase structures. In Proceedings of the first international conference on Human language technology research (pp. 1-5). Association for Computational Linguistics.

Xia, F., Rambow, O., Bhatt, R., Palmer, M., & Misra Sharma, D. (2009). Towards a multi-representational treebank. LOT Occasional Series, 12, 159-170.

محمدحسین دهقان دانشجوی کارشناسی ارشد مهندسی نرم‌افزار در دانشگاه تهران است. وی تحصیلات خود را در مقطع کارشناسی مهندسی نرم‌افزار را در دانشکدهٔ مهندسی کامپیوتر دانشگاه اراک با رتبه یک در سال ۱۳۹۱ به پایان رساند. زمینه‌های پژوهشی مورد علاقهٔ ایشان پردازش هوشمند متن و زبان طبیعی و بازیابی اطلاعات است.
نشانی رایانامهٔ ایشان عبارت است از:
mh.dehghan@ut.ac.ir

هشام فیلی تحصیلات خود را در مقطع کارشناسی مهندسی نرم‌افزار در دانشکدهٔ مهندسی کامپیوتر دانشگاه صنعتی شریف با رتبه یک در سال ۱۳۷۶ به پایان رساند؛ سپس مقاطع کارشناسی ارشد نرم‌افزار و دکترای هوش مصنوعی را به‌ترتیب در سال‌های ۱۳۷۸ و ۱۳۸۵ در همان دانشکده تکمیل کرد. از سال ۱۳۸۷ تاکنون عضو هیئت علمی دانشکده مهندسی برق و کامپیوتر دانشکدهٔ فنی دانشگاه تهران است. زمینه‌های پژوهشی مورد علاقهٔ ایشان پردازش هوشمند متن و زبان طبیعی، ترجمهٔ ماشینی، داده‌کاوی، بازیابی اطلاعات و شبکه‌های اجتماعی هستند.
نشانی رایانامهٔ ایشان عبارت است از:
hfaili@ut.ac.ir