หนึ่งมาพิจารณาในการถอดอักษรเป็นภาษาที่สองด้วย ซึ่งก็เป็นที่นิยมทำกันในการพัฒนาระบบการถอดอักษรภาษาอื่นๆ

แต่ปัญหาใหญ่ของการใช้เสียงอ่านภาษาอังกฤษด้วย คือ ระบบที่พัฒนาจะต้องมีโมดูลที่สามารถถ่ายเสียงคำภาษาอังกฤษออกมาได้ด้วย ซึ่งถือเป็นงานวิจัยต่างหากอีกหนึ่งงานวิจัย ในที่นี้ ผู้วิจัยจึงได้ทดลองสร้างโมดูลสำหรับถ่ายเสียงคำภาษาอังกฤษตามแบบงานวิจัยที่เคยมีผู้ศึกษาไว้แล้ว โดยได้ทดลองทำใน 2 แนวทาง วิธีแรกคือใช้ดีซิชันทรีตามแบบของ Lenzo (1998) และวิธีที่สองคือการใช้ตารางกฎตามแบบของ Bosch and Daelemans (1993) Lenzo (1998) ได้รายงานผลความถูกต้องของการถ่ายเสียงภาษาอังกฤษด้วยการใช้ดีซิชันทรีที่ 95% ในระดับหน่วยเสียงและ 79% ในระดับคำ ในที่นี้ ผู้วิจัยได้ทดลองสร้างดิซิชันทรีตามแบบ Lenzo โดยทดลองกำหนดความลึกหรือจำนวนชั้นการตัดสินใจที่ 3,4,5,6 และ 7 ชั้น และใช้ Carnegie Mellon Pronouncing Dictionary version 0.6 เป็นข้อมูลฝึกสอน ส่วนวิธีการแบบตารางกฎคือวิธีที่ Bosch and Daelemans (1993) ใช้ในการถ่ายเสียงภาษาอังกฤษ (ได้อธิบายไปแล้วในหัวข้อ 2.2.2) ซึ่งรายงานความถูกต้องที่ 95% ในระดับหน่วยเสียง ในที่นี้จะทดลองโดยใช้ข้อมูลฝึกสอนและทดสอบชุดเดียวกับที่ใช้ในการทดสอบดีซิชันทรี โดยจะทดสอบการใช้ตารางกฎขนาด 3 บริบทซ้ายขวา, 4 บริบทซ้ายขวา, 5 บริบทซ้ายขวา, และ 6 บริบทซ้ายขวา (ในที่นี้นั้นจะใช้บริบทซ้ายขวาดังนี้ 0-1, 1-1, 1-2, 2-2, 2-3, 3-3, 3-4, 4-4, 4-5, 5-5, 5-6 และ 6-6 ส่วนตารางคาดเดาจะใช้บริบทซ้ายขวาดังนี้ 0-0, 1-1, 2-2, 3-3 และ 4-4)

ตารางข้างล่างเป็นผลความถูกต้องของการถ่ายเสียงภาษาอังกฤษในคำที่ไม่อยู่ในข้อมูลฝึกสอน เปรียบเทียบระหว่างการใช้ดิซิชันทรีที่ระดับความลึก 4-7 ชั้น และการใช้ตารางกฎที่ใช้บริบทซ้ายขวาขนาด 3-6 ตัวอักษร โดยผลจากการใช้ดีซิชันทรีระดับที่ยอมรับได้จะอยู่ที่ความลึกตั้งแต่ 5 ขึ้นไปโดยที่ความลึกระดับ 7 จะให้ผลไม่แตกต่างจากระดับความลึก 6 เท่าใดนัก ส่วนวิธีการใช้ตารางกฎนั้น เมื่อพิจารณาบริบทมากขึ้น ผลก็จะดีมากขึ้น เมื่อเปรียบเทียบระหว่างสองวิธี ดีซิชันทรีดูจะให้ผลที่ดีกว่าการใช้ตารางกฎ แต่การตัดสินว่าวิธีใดจะเหมาะสมกว่ากัน ในที่นี้จำเป็นต้องพิจารณาผลจากการนำไปใช้ร่วมกับโปรแกรมถอดอักษรภาษาอังกฤษเป็นไทย ซึ่งจะนำเสนอต่อไป

	ระดับคำ			ระดับอักขระ					
ดีซิชันทรี	ถูก	ผิด	%ถูก	ถูก	ผิด	%ถูก	ระยะห่าง	จำนวนอักษร	%ใกล้เคียง
ความลึก=4	5591	5448	50.6%	72556	9094	88.9%	8871	81650	89.1%
ความลึก=5	6163	4876	55.8%	73757	7893	90.3%	7712	81650	90.6%
ความลึก=6	6289	4750	57.0%	74040	7610	90.7%	7439	81650	90.9%
ความลึก=7	6287	4752	57.0%	74035	7615	90.7%	7443	81650	90.9%
ตารางกฎ	ถูก	ผิด	%ถูก	ถูก	ผิด	%ถูก	ระยะห่าง	จำนวนอักษร	%ใกล้เคียง
บริบท=3	5960	5079	54.0%	73548	8102	90.1%	7915	81650	90.3%
บริบท=4	6170	4869	55.9%	73842	7808	90.4%	7623	81650	90.7%
บริบท=5	6233	4806	56.5%	73906	7744	90.5%	7550	81650	90.8%
บริบท=6	6262	4777	56.7%	73932	7718	90.5%	7521	81650	90.8%

ตาราง 10: ผลการเปรียบเทียบโมดูลถ่ายเสียงภาษาอังกฤษแบบต่างๆ

3.2.8 ระบบที่ใช้ตารางกฎของตัวอักษรและเสียงอ่าน

จากเหตุผลเรื่องความสำคัญของเสียงอ่านในการถอดอักษรที่ได้กล่าวมาแล้ว ในงานวิจัยนี้ จึงได้นำระบบในข้อ 3.2.1 มาพัฒนาเพิ่มเติมโดยใช้ข้อมูลทั้งตัวเขียนและเสียงอ่านของภาษาที่หนึ่งมาพิจารณาพร้อมกันในการถอดอักษรเป็นภาษาที่สอง แต่ไม่ได้พัฒนาออกมาเป็นระบบแบบประมวลผลต่อเนื่อง (pipeline) ที่ถ่ายคำภาษาอังกฤษเป็นเสียงอ่านก่อนจะแปลงเป็นตัวอักษรไทย ตามแบบที่ใช้กันโดยทั่วไปใน แนวทางใหม่นี้อาศัยการพิจารณาทั้งตัวเขียนและเสียงอ่านที่เกิดขึ้นในภาษาอังกฤษไปพร้อมกัน ทั้งนี้เพราะผู้วิจัยเห็นว่าน่าจะสอดคล้องกับลักษณะการวางเกณฑ์ของทางราชบัณฑิตยสถานมากกว่า และมีข้อดีมากกว่าการประมวลผลแบบต่อเนื่อง เนื่องจากผลการถ่ายเสียงภาษาอังกฤษจากโมดูลที่สร้างขึ้นนั้นไม่ได้ถูกต้องสมบูรณ์แต่อยู่ที่ประมาณ 57% ในระดับคำ และ 91% ในระดับหน่วยเสียง หากประมวลผลแบบต่อเนื่องจากรูปอักษรเป็นเสียงภาษาอังกฤษก่อนจะถอดเป็นอักษรไทย ผลลัพธ์ที่ได้มีแนวโน้มที่จะผิดทันทีหากพิจารณาเฉพาะเสียงที่ถ่ายออกมา การพิจารณาทั้งรูปและเสียงที่ถ่ายอาจช่วยลดความผิดพลาดที่อาจจะเกิดขึ้นได้ ดังนั้น ตารางกฎที่ใช้จึงแตกต่างจากระบบแรกที่พัฒนา โดยกฎที่ใช้จะมีข้อมูลทั้งรูปอักษรและเสียงอ่าน เช่น i - /aI/ -> ไ หมายถึง ตัวอักษร i ที่ออกเสียง /aI/ จะถอดเป็น ไ และเนื่องจากเสียงอ่านสามารถสร้างโดยวิธีการใช้ตารางกฎหรือการใช้ดีซิชันทรีตามที่กล่าวมาแล้ว ในที่นี้จึงทดสอบระบบนี้ทั้งสองลักษณะ คือ ทดสอบโปรแกรมถอดอักษรอังกฤษเป็นไทยแบบตารางกฎที่พิจารณารูปและเสียงอ่านที่ได้จากการใช้ตารางกฎที่ดูบริบทซ้ายขวา 6 ตำแหน่ง และแบบตารางกฎที่พิจารณารูปและเสียงอ่านที่ได้จากการใช้ดีซิชันทรีที่ความลึก 7 ผลของการทดสอบโปรแกรมทั้ง 2 แบบแสดงในตารางข้างล่างนี้

	ระดับคำ			ระดับอักขระ					
	ถูก	ผิด	%ถูก	ถูก	ผิด	%ถูก	ระยะห่าง	จำนวนอักษร	%ใกล้เคียง
ชุดที่ 1	1061	575	64.9%	10246	982	91.3%	1325	11228	88.2%
ชุดที่ 2	1090	546	66.6%	10321	947	91.6%	1260	11268	88.8%
ชุดที่ 3	1040	596	63.6%	10355	1008	91.1%	1353	11363	88.1%
ชุดที่ 4	1076	560	65.8%	10208	965	91.4%	1308	11173	88.3%
ชุดที่ 5	1074	563	65.6%	10346	924	91.8%	1255	11270	88.9%
เฉลี่ย	1068.2	568	65.3%	10295.2	965.2	91.4%	1300.2	11260.4	88.5%
รวมทุกชุด	8153	28	99.7%	56268	34	99.9%	48	56302	99.9%

ตาราง 11: ผลจากระบบตารางกฎที่ใช้ตัวเขียนและเสียงอ่านที่ได้จากการใช้ตารางกฎ

	ระดับคำ			ระดับอักขระ					
	ถูก	ผิด	%ถูก	ถูก	ผิด	%ถูก	ระยะห่าง	จำนวนอักษร	%ใกล้เคียง
ชุดที่ 1	1074	562	65.6%	10251	977	91.3%	1312	11228	88.3%
ชุดที่ 2	1073	563	65.6%	10327	941	91.6%	1299	11268	88.5%
ชุดที่ 3	1057	579	64.6%	10365	998	91.2%	1342	11363	88.2%
ชุดที่ 4	1065	571	65.1%	10195	978	91.2%	1326	11173	88.1%
ชุดที่ 5	1065	572	65.1%	10327	943	91.6%	1276	11270	88.7%
เฉลี่ย	1066.8	569.4	65.2%	10293	967.4	91.4%	1311	11260.4	88.4%
รวมทุกชุด	8148	33	99.6%	56262	40	99.9%	55	56302	99.9%

ตาราง 12: ผลจากระบบตารางกฎที่ใช้ตัวเขียนและเสียงอ่านที่ได้จากการใช้ดีซิชันทรี

ผลที่ได้จากระบบทั้งสองแบบนั้นใกล้เคียงกัน โดยระบบที่ใช้เสียงอ่านจากตารางกฎจะให้ค่าความถูกต้องทั้งในระดับคำและระดับอักขระดีกว่าระบบที่ใช้เสียงอ่านจากดีซิชันทรีเล็กน้อย โดยระบบที่ใช้เสียงอ่านภาษาอังกฤษจากตารางกฎสามารถถอดอักษรคำที่ไม่เคยเห็นได้ถูกต้อง 65.3% ในระดับคำ และ 91.4% ในระดับอักขระ โดยมีความถูกต้องเมื่อเทียบกับคำเป้าหมายที่ 88.5% ส่วนในคำที่เคยเห็นแล้วจะถอดอักษรได้ถูกต้อง 99.7% ในระดับคำ และ 99.9% ในระดับอักขระ โดยมีค่าความใกล้เคียงกับเป้าหมายที่ 99.9%

นอกจากนี้ ผู้วิจัยยังได้ทดลองเทียบเสียงอ่านภาษาอังกฤษที่สร้างขึ้นเป็นเสียงภาษาไทย (ดูภาคผนวก xx ตารางเทียบเสียงภาษาอังกฤษเป็นไทย) ก่อนที่จะนำไปพิจารณาพร้อมกับรูปอักษรภาษาอังกฤษในการสร้างตารางกฎการถอดอักษร โดยทดลองกับระบบที่ใช้เสียงอ่านภาษาอังกฤษที่ได้จากตารางกฎ ผลการทดสอบการแปลงเป็นเสียงภาษาไทยก่อน ในระบบนี้ให้ผลไม่แตกต่างจากการใช้เสียงอ่านภาษาอังกฤษเท่าใดนัก โดยให้ความถูกต้องที่ 65.3% ในระดับคำ และ 91.4% ในระดับตัวอักษรเช่นกัน

	ระดับคำ			ระดับอักขระ					
	ถูก	ผิด	%ถูก	ถูก	ผิด	%ถูก	ระยะห่าง	จำนวนอักษร	%ใกล้เคียง
ชุดที่ 1	1060	576	64.8%	10244	984	91.2%	1334	11228	88.1%
ชุดที่ 2	1089	547	66.6%	10320	948	91.6%	1267	11268	88.8%
ชุดที่ 3	1042	594	63.7%	10359	1004	91.2%	1353	11363	88.1%
ชุดที่ 4	1080	556	66.0%	10206	967	91.3%	1319	11173	88.2%
ชุดที่ 5	1073	564	65.5%	10347	923	91.8%	1252	11270	88.9%
เฉลี่ย	1068.8	567.4	65.3%	10295.2	965.2	91.4%	1305	11260.4	88.4%
รวมทุกชุด	8152	29	99.6%	56267	35	99.9%	50	56302	99.9%

ตาราง 13: ทดสอบระบบที่ใช้ตัวเขียนและเสียงอ่านที่ได้จากการใช้ตารางกฎแต่แปลงเป็นไทย

3.2.9 ระบบที่ใช้ดีซิชันทรีของรูปอักษรและเสียงอ่าน

การทดลองส่วนนี้ เป็นการนำระบบถอดอักษรที่ใช้วิธีการของดีซิชันทรีในข้อ 2.2.3 มาปรับปรุงโดยเพิ่มเสียงอ่านในการพิจารณาสร้างดีซิชันทรี ซึ่งเสียงอ่านนี้สามารถสร้างได้จากวิธีการใช้ตารางกฎหรือการใช้ดีซิชันทรีดังที่กล่าวมาแล้ว ดังนั้น จึงทำการทดลองเป็นสองส่วน โดยส่วนแรกเป็นการสร้างดีซิชันทรีที่ใช้รูปและเสียงอ่านภาษาอังกฤษที่ได้จากโมดูลถ่ายเสียงแบบดีซิชันทรี และส่วนที่สองเป็นการสร้างดีซิชันทรีที่ใช้รูปและเสียงอ่านภาษาอังกฤษที่ได้จากโมดูลถ่ายเสียงแบบตารางกฎ ผลการทดสอบแสดงให้เห็นในตารางข้างล่างนี้

	ระดับคำ			ระดับอักขระ					
	ถูก	ผิด	%ถูก	ถูก	ผิด	%ถูก	ระยะห่าง	จำนวนอักษร	%ใกล้เคียง
ชุดที่ 1	1054	582	64.4%	10261	967	91.4%	1362	11228	87.9%
ชุดที่ 2	1069	567	65.3%	10310	958	91.5%	1312	11268	88.4%
ชุดที่ 3	1048	588	64.1%	10373	990	91.3%	1376	11363	87.9%
ชุดที่ 4	1055	581	64.5%	10187	986	91.2%	1343	11173	88.0%
ชุดที่ 5	1039	598	63.5%	10280	990	91.2%	1374	11270	87.8%
เฉลี่ย	1053	583.2	64.4%	10282.2	978.2	91.3%	1353.4	11260.4	88.0%
รวมทุกชุด	7979	202	97.5%	56053	249	99.6%	314	56302	99.4%

ตาราง 14: ทดสอบระบบดีซิชันทรีที่ใช้ตัวเขียนและเสียงอ่านที่ได้จากการใช้ดีซิชันทรี

	ระดับคำ			ระดับอักขระ					
	ถูก	ผิด	%ถูก	ถูก	ผิด	%ถูก	ระยะห่าง	จำนวนอักษร	%ใกล้เคียง
ชุดที่ 1	1056	580	64.5%	8895	2351	79.1%	1335	11228	88.1%
ชุดที่ 2	1009	627	61.7%	8901	2477	78.2%	1524	11268	86.5%
ชุดที่ 3	1052	584	64.3%	8753	2392	78.5%	1436	11363	87.4%
ชุดที่ 4	1024	612	62.6%	8989	2336	79.4%	1435	11173	87.2%
ชุดที่ 5	1011	626	61.8%	8892	2316	79.3%	1377	11270	87.8%
เฉลี่ย	1030.4	605.8	63.0%	8886	2374.4	78.9%	1421.4	11260.4	87.4%
รวมทุกชุด	7968	213	97.4%	48606	7696	86.3%	345	56302	99.4%

ตาราง 15: ทดสอบระบบดีซิชันทรีที่ใช้ตัวเขียนและเสียงอ่านที่ได้จากการใช้ตารางกฎ

จากผลการทดสอบ พบว่าระบบที่ใช้โมดูลถ่ายเสียงภาษาอังกฤษแบบดีซิชันทรีให้ผลดีกว่าเมื่อนำมาใช้ร่วมกับระบบแบบดีซิชันทรี สำหรับคำที่ไม่เคยเห็นมาก่อน ให้ความถูกต้องที่ 64.4% ในระดับคำ และ 91.3% ในระดับตัวอักษร โดยมีค่าความใกล้เคียงเป้าหมายที่ 88% สำหรับคำที่เคยเห็นมาแล้ว จะได้ความถูกต้อง 97.5% ในระดับคำ และ 99.6% ในระดับตัวอักษร โดยใกล้เคียงกับเป้าหมาย 99.4% และเมื่อทดสอบระบบดีซิชันทรี โดยใช้โมดูลถ่ายเสียงภาษาอังกฤษแบบดีซิชันทรีแล้วจึงแปลงเป็นเสียงภาษาไทย (ดูตารางผลข้างล่าง) ผลปรากฏว่า ค่าความถูกต้องสูงขึ้นเล็กน้อยเป็น 64.8% ระดับคำ 91.4% ระดับตัวอักษร ความใกล้เคียงเป้าหมายอยู่ที่ 88.1% ในคำที่ไม่เคยเห็น ส่วนคำที่เคยเห็นมาแล้ว ได้ความถูกต้องในระดับคำที่

97.5% ในระดับอักขระที่ 99.5% ค่าความใกล้เคียงเป้าหมายที่ 99.4% แต่ผลที่ได้นี้ก็ยังต่ำกว่าระบบที่ใช้ตารางกฎที่กล่าวมาแล้วโดยเฉพาะในกรณีของคำที่เคยเห็นมาก่อน (ดู 3.2.8) ระบบที่ใช้ตารางกฎสำหรับถ่ายเสียงภาษาอังกฤษและถอดอักษรอังกฤษเป็นไทยจึงเหมาะสมสำหรับงานนี้มากกว่าระบบที่ใช้ดีซิชันทรี

	ระดับคำ			ระดับอักขระ					
	ถูก	ผิด	%ถูก	ถูก	ผิด	%ถูก	ระยะห่าง	จำนวนอักษร	%ใกล้เคียง
ชุดที่ 1	1065	571	65.1%	10268	960	91.4%	1345	11228	88.0%
ชุดที่ 2	1060	576	64.8%	10298	970	91.4%	1347	11268	88.0%
ชุดที่ 3	1055	581	64.5%	10388	975	91.4%	1358	11363	88.0%
ชุดที่ 4	1071	565	65.5%	10217	956	91.4%	1303	11173	88.3%
ชุดที่ 5	1050	587	64.1%	10308	962	91.5%	1328	11270	88.2%
เฉลี่ย	1060.2	576	64.8%	10295.8	964.6	91.4%	1336.2	11260.4	88.1%
รวมทุกชุด	7975	206	97.5%	56037	265	99.5%	318	56302	99.4%

ตาราง 16: ทดสอบระบบดีซิชันทรีที่ใช้ตัวเขียนและเสียงอ่านที่ได้จากการใช้ดีซิชันทรีแล้วแปลงเป็นเสียงไทย

3.2.10 ระบบที่ใช้ข้อมูลทั้งสองภาษาพร้อมเสียงอ่านและมองแบบเป็นกลุ่มก้อน (chunk-based)

การทดลองส่วนนี้ เป็นการนำระบบในข้อ 3.2.6 มาพัฒนาต่อโดยใช้แบบจำลองทางสถิติในลักษณะเดิม แต่ผนวกเอาหน่วยเสียงที่ได้จากโมดูลถ่ายเสียงภาษาอังกฤษประกอบการพิจารณาด้วย (ผู้วิจัยไม่ได้ทดลองกับระบบ 3.2.4 และ 3.2.5 เนื่องจากเป็นระบบในลักษณะคล้ายกันกับ 3.2.6 แต่ให้ผลความถูกต้องที่ต่ำที่สุดในการทดลอง) แบบจำลองทางสถิติที่ใช้เป็นดังนี้

$$\arg\max_T P(T \mid E) = \arg\max_T \frac{P(T,E)}{P(E)} = \arg\max_T P(T,E)$$

$$= \arg\max_T P(t_1 t_2 ..t_n, e_1 ph_1 e_2 ph_2 ...e_n ph_n)$$

$$= \arg\max_T P(< t_{1..a}, e_{1..a} ph_{1..a} >, < t_{a+1..b}, e_{a+1..b} ph_{a+1..b} > .... < t_{m+1..n}, e_{m+1..n} ph_{m+1..n} >)$$

$$= \arg\max P(< ct_1, ce_1 cph_1 >, < ct_2, ce_2 cph_2 >, ... < ct_n, ce_n cph_n >)$$

$$\approx \prod_{i=1}^{n} P(< ct_i, ce_i cph_i > | < ct_{i-2}, ce_{i-2} cph_{i-2} >< ct_{i-1}, ce_{i-1} cph_{i-1} >)$$

(ct_i *and* ce_i, cph_i *are a chunck of Thai and English characters with its corresponding sounds*)

โดยที่ $t_1 t_2 ...t_n$ คือสายอักขระของคำไทยที่ได้ $e_1 ph_1 e_2 ph_2 ... e_n ph_n$ คือสายอักขระภาษาอังกฤษรวมกับหน่วยเสียงที่ได้จากการถ่ายเสียง $t_{i..j} e_{i..j} ph_{i..j}$ คือกลุ่มของสายอักขระไทยที่เข้าคู่กับกลุ่มของสายขระอังกฤษรวมกับหน่วยเสียง ความยาวของกลุ่มอักขระนี้เป็นได้ตั้งแต่หนึ่งตัวอักษรอังกฤษไปจนถึงความยาวทั้งคำ ในที่นี้ได้ทดลองสองแบบ แบบแรกใช้หน่วยเสียงภาษาอังกฤษที่ถ่ายได้ แบบที่สองใช้หน่วยเสียงที่แปลงเทียบเป็นไทยแล้ว ผลการทดสอบแสดงในตารางข้างล่าง

	ระดับคำ			ระดับอักขระ		
	ถูก	ผิด	%ถูก	ระยะห่าง	จำนวนอักษร	%ใกล้เคียง
ชุดที่ 1	1105	531	67.5%	1295	11228	88.5%
ชุดที่ 2	1116	520	68.2%	1264	11268	88.8%
ชุดที่ 3	1109	527	67.8%	1338	11363	88.2%
ชุดที่ 4	1113	523	68.0%	1242	11173	88.9%
ชุดที่ 5	1125	512	68.7%	1214	11270	89.2%
เฉลี่ย	1113.6	522.6	68.1%	1270.6	11260.4	88.7%
รวมทุกชุด	8167	14	99.8%	30	56302	99.9%

ตาราง 17 : ผลจากระบบที่จับคู่อักษรเป็นกลุ่มและใช้เสียงอ่านภาษาอังกฤษ

	ระดับคำ			ระดับอักขระ		
	ถูก	ผิด	%ถูก	ระยะห่าง	จำนวนอักษร	%ใกล้เคียง
ชุดที่ 1	1106	530	67.6%	1301	11228	88.4%
ชุดที่ 2	1114	522	68.1%	1264	11268	88.8%
ชุดที่ 3	1108	528	67.7%	1324	11363	88.3%
ชุดที่ 4	1119	517	68.4%	1228	11173	89.0%
ชุดที่ 5	1129	508	69.0%	1214	11270	89.2%
เฉลี่ย	1115.2	521	68.2%	1266.2	11260.4	88.8%
รวมทุกชุด	8166	15	99.8%	32	56302	99.9%

ตาราง 18 : ผลจากระบบที่จับคู่อักษรเป็นกลุ่มและใช้เสียงอ่านภาษาไทย

จากผลการทดลอง พบว่าในระบบที่มองคู่อักษรภาษาอังกฤษและไทยแบบเป็นกลุ่มนี้ แบบที่ใช้เสียงอ่านภาษาไทยให้ผลที่ดีกว่าเล็กน้อย คือในคำที่ไม่เคยเห็นมาก่อน เมื่อนำเสียงที่แปลงให้เข้ากับภาษาไทยมาพิจารณาด้วยจะให้ผลที่ถูกต้อง 68.2% ในระดับคำ โดยมีค่าความใกล้เคียงกับเป้าหมายที่ 88.7% และเมื่อนำเสียงอ่านภาษาอังกฤษมาพิจารณาจะให้ผลที่ถูกต้อง 68.1% ในระดับคำ โดยมีค่าความใกล้เคียงกับเป้าหมายที่ 88.8% ซึ่งผลที่ได้นี้ถือว่าสูงที่สุด สูงกว่าวิธีการแบบตารางกฎที่ใช้รูปอักษรและเสียงอ่าน (65.3%) อย่างไรก็ตาม ระบบนี้ก็ข้อจำกัดในเรื่องของการใช้ทรัพยากรและเวลาในการประมวลผลมากกว่าวิธีการอื่นๆ

3.2.10 การพัฒนาโปรแกรมถอดอักษรอังกฤษเป็นไทยแบบใช้กฎที่สร้างเอง

จากผลที่ได้จากการทดลองระบบต่างๆ ซึ่งให้ผลที่ไม่สูงอย่างที่ต้องการ (>97%) ซึ่งเมื่อตรวจสอบคำที่ถอดอักษรไม่ถูกต้อง พบว่ามีจำนวนหนึ่งที่ถอดออกมาแล้วไม่เป็นภาษา เช่น diabase – ไดาเบส, dialysis – ไแดลิซิส, santiago – ซานเดียาโก ซึ่งหากเราถอดอักษรด้วยตัวเอง แม้ว่าอาจไม่ตรงตามเกณฑ์ของราชบัณฑิตยสถานทุกประการ แต่ก็คงไม่สร้างคำที่ผิดจากอักขรวิธีภาษาไทยเช่นนี้ จึงเกิดคำถามว่า หากพัฒนาเป็นระบบที่ใช้กฎการถอดอักษรที่เขียนขึ้นเองโดยไม่ใช้กฎที่เครื่องสร้างขึ้นจากข้อมูลฝึกสอน ผลจะแตกต่างกันหรือดีขึ้นหรือไม่ ในที่นี้ผู้วิจัยจึงได้ทดลองพัฒนาระบบถอดอักษรอีกระบบที่อาศัยกฎการถอดอักษร

ที่สร้างขึ้น โดยอาศัยตารางเทียบอักษรของราชบัณฑิตยสถานเป็นสำคัญ โดยใช้กฎที่พิจารณาทั้งรูปอักษรและเสียงอ่านที่ปรับเป็นเสียงไทยแล้วเป็นหลักเพราะจะสะดวกต่อการเทียบกับตารางของราชบัณฑิตยสถานมากกว่า ตัวอย่างของกฎที่สร้างขึ้น เช่น l-/l/ -> ล, a-/a/ -> ะ, ch-/kh/ -> ค, iu-/iia/ -> เีย, er-/r/ -> เ.อร์, eo-/iia/ -> เีย เป็นต้น นอกจากจะใช้ตัวอย่าง 314 คำจากตารางกฎของราชบัณฑิตยสถาน ผู้วิจัยได้นำคำจำนวนหนึ่งอีกประมาณ 300 คำมาทดสอบเพื่อดูความถูกต้องของกฎ หากยังไม่ครอบคลุมก็จะปรับแก้ หรือเพิ่มเติมกฎเข้าไปอีก ทำเช่นนี้ไปเรื่อยๆ จนกว่าจะได้ผลเป็นที่พอใจ และเนื่องจากเสียงอ่านที่ได้จากโมดูลถ่ายเสียงภาษาอังกฤษอาจไม่สอดคล้องกับลำดับตัวอักษรของคำในทุกกรณีหรือได้เสียงที่ไม่ถูกต้อง กฎที่เขียนขึ้นบางกฎจึงอาจดูแปลก แต่จำเป็นสำหรับระบบหากต้องการให้ถอดอักษรได้ถูกต้อง ตัวอย่างเช่น เมื่อถ่ายเสียงคำว่า marx เสียงที่ได้คือ /m-aa-kh-kh/ ดังนั้น เพื่อให้ถอดอักษรคำนี้ได้ จึงต้องมีกฎ r-/kh/ -> ร หรือคำว่า cullan ซึ่งถอดเสียงได้ /kh-a-l-a-#-n/ จึงต้องมีกฎ l-/a/ -> ล เป็นต้น

การสร้างกฎเองนั้นค่อนข้างยุ่งยากและใช้เวลานานมากสำหรับการตรวจสอบแก้ไขผลจากตัวอย่างทีละคำทีละคำ เพราะการเพิ่มกฎเพื่อให้เหมาะสำหรับคำบางคำ อาจส่งผลให้ถอดอักษรคำอื่นที่เคยตรวจสอบมาก่อนให้ผลที่ผิดไปจากเดิมได้ จึงควบคุมผลให้ได้ถูกต้องอย่างที่ต้องการยาก ผลการทดสอบระบบที่สร้างขึ้นซึ่งมีจำนวนกฎทั้งสิ้น 266 กฎ พบว่า ระบบสามารถถอดอักษรได้ถูกต้องเพียง 27.41% ในระดับคำ ดังนั้น แนวทางการพัฒนาลักษณะนี้จึงไม่ใช่ทางเลือกที่ต้องการ

3.3 การประเมินผลโปรแกรมถอดอักษรอังกฤษเป็นไทย

จากผลการทดสอบระบบต่างๆ พบว่า การใช้โมดูลถ่ายเสียงภาษาอังกฤษประกอบการถอดอักษรภาษาอังกฤษเป็นไทย ทำให้ได้ผลดีขึ้นในทุกระบบที่ทดสอบ โดยระบบที่ให้ผลดีที่สุด คือ ระบบที่จับคู่รูปอักษรทั้งสองภาษาพร้อมกับเสียงอ่านแบบเป็นกลุ่มก้อน (68.2%) รองลงไปคือ ระบบที่ใช้ตารางกฎและใช้ตารางกฎของรูปและเสียงอ่าน (65.3%) แต่อย่างไรก็ตาม เมื่อพิจารณาประสิทธิภาพของการถอดอักษรคำที่ไม่เคยเห็นมาก่อน ความถูกต้องในระดับคำก็ยังไม่น่าพอใจ คือ สูงขึ้นมาเพียง 68.2% และ 65.3% ตามลำดับ และถูกต้อง 91.4% ในระดับตัวอักษร การเพิ่มเพียงข้อมูลเสียงอ่านให้กับโปรแกรมนั้นแม้จะทำให้ได้ผลการทำงานที่ดีขึ้น แต่ก็ยังไม่เพียงพอกับที่ต้องการ จำเป็นต้องพัฒนาให้มีประสิทธิภาพมากขึ้นอีก ซึ่งต้องอาศัยการวิเคราะห์ข้อผิดพลาดที่เกิดขึ้นว่า มีสาเหตุใดบ้างที่ทำให้โปรแกรมไม่สามารถถอดอักษรได้ถูกต้องตามเกณฑ์ของราชบัณฑิตยสถาน

3.3.1 ความหลากหลายของการออกเสียง

ปัญหาหนึ่งที่สังเกตพบคือ เรื่องของความหลากหลายในการออกเสียง วัลย์วรา (2548) ได้ตั้งข้อสังเกตว่า คำเดียวกันอาจถอดอักษรเป็นไทยได้หลายแบบ เช่น คำว่า leonard พบว่ามีการทับศัพท์เป็นคำไทยได้หลายแบบ ได้แก่ เลนาร์ด เลนเนิร์ด เลียวนาร์ด ซึ่งอาจจะเนื่องมาจากการที่ผู้ทับศัพท์ออกเสียงคำนี้ด้วยสำเนียงที่แตกต่างกัน แม้ว่าในระหว่างการรวบรวมข้อมูลเพื่อสร้างคลังคำทับศัพท์ คำที่ทับศัพท์แตกต่างกันนี้จะถูกคัดมาเพียงคำเดียวโดยผู้วิจัยเลือกคำที่เห็นว่าตรงตามเกณฑ์ของราชบัณฑิตยสถานมากที่สุด แต่ความแตกต่างนี้สะท้อนให้เห็นว่าในการทับศัพท์จริง ผู้ทับศัพท์อาจทับศัพท์ด้วยสำเนียงที่แตกต่างกัน ข้อมูลคลังทับศัพท์ที่รวบรวมมาจึงอาจเป็นการทับศัพท์ที่ใช้สำเนียงแตกต่างกันตามความคุ้นเคยของผู้เขียน ผลจากความหลากหลายของการออกเสียงนี้ จึงส่งผลให้คอมพิวเตอร์ไม่สามารถสร้างกฎการถอดอักษรได้อย่างแม่นยำ

นอกจากนี้ คำภาษาอังกฤษเองนั้น ก็มีการเปลี่ยนแปลงมาตลอด มีการยืมคำจากภาษาอื่นๆเข้ามา จึงทำให้ความสัมพันธ์ระหว่างรูปเขียนและเสียงนั้นไม่คงที่ตายตัว จึงค่อนข้างยากที่จะกำหนดกฎการออกเสียงภาษาอังกฤษที่คงที่หรือพิจารณาจากตัวอักษรข้างเคียงได้เสมอ ตัวอย่างเช่น คำว่า peirce - เพิร์ซ และ peirse – เพียร์ส มีรูปคำที่ค้ายคลึงกันมากต่างกันเพียงตัวสะกด c และ s แต่ทับศัพท์ด้วยเสียงสระที่แตกต่างกันเป็น เออ และ เอีย ตามลำดับ, คำว่า yardley - ยาร์ดเลย์ และ key - คีย์ มีส่วนท้ายที่เหมือนกันคือ ey แต่ทับศัพท์ด้วยเสียงที่แตกต่างกันเป็น เอ และ อี ตามลำดับ เป็นต้น

ในกรณีที่มีหลายๆทางเลือก จากความหลากหลายนี้ โปรแกรมถอดอักษรแบบตารางกฎก็จะพิจารณาทางเลือกที่พบมากกว่า ซึ่งจะส่งผลให้คำตอบส่วนใหญ่น่าจะถูกต้องตรงตามข้อมูลที่พบ แต่ก็จะมีผลให้คำตอบบางส่วนไม่ถูกต้องคือไม่ตรงกับข้อมูลที่มี จึงทำให้มีข้อผิดพลาดอย่างหลีกเลี่ยงไม่ได้ คำถามที่ตามมาคือ คำที่โปรแกรมถอดอักษรแล้วไม่ตรงกับข้อมูลที่พบ จำเป็นต้องเป็นการถอดอักษรที่ผิดพลาดหรือไม่ คำตอบคือ ไม่เสมอไป เพราะโปรแกรมอาจเลือกถอดอักษรในแบบที่ไม่ตรงกับรูปคำที่พบในข้อมูล แต่ก็อาจเป็นรูปแปรที่ยอมรับได้ว่าถูกต้องตามเกณฑ์ของราชบัณฑิตยสถานได้เช่นกัน ตัวอย่างเช่น โปรแกรมอาจถอดอักษร trumpet เป็น ทรัมเพต แต่ข้อมูลที่ใช้จริงเป็น ทรัมเป็ต กรณีนี้ โปรแกรมถอดอักษร p เป็น พ ตามหลักการเทียบพยัญชนะได้ถูกต้อง เพียงแต่คำนี้เป็นคำที่ทับศัพท์มากก่อนโดยใช้ ป , ซึ่งราชบัณฑิตยสถานกำหนดไว้ว่า "คำทับศัพท์ที่ใช้กันมานานจนถือเป็นภาษาไทย และปรากฏในพจนานุกรม ฉบับราชบัณฑิตยสถานแล้ว ให้ใช้ต่อไปตามเดิม" ดังนั้น การประเมินผลการทำงานของโปรแกรมจึงต้องพิจารณาทุกรายคำที่ได้ผลไม่ตรงกับข้อมูลว่าเป็นการถอดอักษรที่ยอมรับได้ตามเกณฑ์ไม่ได้เป็นคำเก่าที่ทับศัพท์มาก่อนหรือไม่ด้วย

3.3.2 ความสอดคล้องของข้อมูลคำทับศัพท์กับเกณฑ์การทับศัพท์ของราชบัณฑิตยสถาน

นอกจากเรื่องความหลากหลายของการออกเสียงแล้ว ก็อาจมีข้อสงสัยว่า ในข้อมูลที่รวบรวมมาเพื่อฝึกสอนให้โปรแกรมนั้น ข้อมูลที่นำมาใช้เป็นไปตามเกณฑ์การทับศัพท์จริงหรือไม่ เพื่อตรวจสอบว่า ข้อมูลคลังทับศัพท์ที่นำมาใช้เป็นพื้นฐานให้คอมพิวเตอร์สร้างกฎการถอดอักษรนั้น ได้มาตรฐานเป็นไปตามเกณฑ์การทับศัพท์ของราชบัณฑิตยสถาน ในส่วนนี้จึงได้ประเมินการทับศัพท์ภาษาอังกฤษที่พบจริงว่ามีความสอดคล้องกับเกณฑ์การทับศัพท์ของราชบัณฑิตยสถานมากน้อยเพียงใด โดยข้อมูลที่ใช้เป็นข้อมูลคำทับศัพท์ที่รวบรวมจากเอกสารสิ่งพิมพ์ที่จัดทำโดยราชบัณฑิตยสถานเองซึ่งก็คือคลังข้อมูลคำทับศัพท์ของราชบัณฑิตยสถานที่นำมาใช้ในการพัฒนาโปรแกรมถอดอักษรภาษาอังกฤษเป็นไทยจำนวน 8,181 คำ โดยผู้วิจัยคาดว่าข้อมูลจะมีความสอดคล้องกับเกณฑ์ของทางราชบัณฑิตยสถานมากเกือบ 100% ผลจากการศึกษาในเรื่องนี้ จะทำให้ทราบถึงปัญหาของการทับศัพท์ภาษาอังกฤษได้ชัดเจนมากขึ้นว่ามีมากน้อยเพียงใด และข้อมูลตัวอักษรใดที่มีปัญหาไม่สอดคล้องกับเกณฑ์มาก

วิธีการที่ใช้ประเมินความสอดคล้องในที่นี้ จะใช้วิธีดูข้อมูลตัวอักษรระหว่างภาษาอังกฤษและภาษาไทยที่ถูกจับคู่ไว้ เทียบกับเกณฑ์การถอดอักษรที่ราชบัณฑิตยสถานวางไว้ เพื่อตรวจสอบว่าได้มีการถอดอักษรตามเกณฑ์ที่วางไว้มากน้อยเพียงใด กล่าวคือ จากข้อมูลที่มีการจับคู่ตัวอักษรระหว่างภาษาอังกฤษและไทยนี้ ผู้วิจัยได้จัดเรียงข้อมูลตามตัวอักษรอังกฤษและตัวอักษรไทยที่ถอดออกมาเป็นกลุ่ม ทั้งนี้เพื่อความสะดวกในการวิเคราะห์ ดังในตัวอย่างข้างล่าง ซึ่งอักษร a ในคำเหล่านี้ถูกถอดเป็น อะ เมื่อจัดเรียงข้อมูลเช่นนี้จะได้คู่ตัวอักษรอังกฤษ-ไทยในการพิจารณาทั้งสิ้น 36,537 คู่ตัวอักษร

	a	mman	x=
centlgr	a	mme	x=
decagr	a	mme	x=
decigr	a	mme	x=
gr	a	mme	x=
hectogr	a	mme	x=
kilogr	a	mme	x=

จากนั้นจึงเปรียบเทียบกับเกณฑ์ของราชบัณฑิตยสถาน โดยตีความในลักษณะที่คาดว่าคอมพิวเตอร์จะกระทำหากยึดตามเกณฑ์ที่ระบุไว้เท่านั้น หากคิดว่าข้อมูลที่พบจริงน่าจะแตกต่างจากที่คอมพิวเตอร์ถอดตามเกณฑ์ก็จะทำเครื่องหมายพิเศษไว้หน้าตัวอย่างนั้น เมื่อวิเคราะห์ข้อมูลทั้งหมดแล้วจึงนับเปอร์เซ็นต์ความสอดคล้องของแต่ละตัวอักษร ตารางข้างล่างแสดงผลเมื่อเทียบจากเกณฑ์การถอดอักษรของราชบัณฑิตยสถาน โดยแจกแจงจำนวนการถอดอักษรที่เป็นไปเกณฑ์ของราชบัณฑิตยสถานและที่แตกต่างไปจากเกณฑ์

ตัวอักษร	สอดคล้อง		ไม่สอดคล้อง	
a	2847	98.2%	51	1.8%
aa	8	88.9%	1	11.1%
ae	25	92.6%	2	7.4%
aea	2	100.0%		0.0%
aer	1	100.0%		0.0%
aeu	1	100.0%		0.0%
ai	69	100.0%		0.0%
aii		0.0%	1	100.0%
air	7	100.0%		0.0%
ao	2	100.0%		0.0%
ar	381	99.7%	1	0.3%
are	13	92.9%	1	7.1%
au	78	97.5%	2	2.5%
aw	16	100.0%		0.0%
ay	46	88.5%	6	11.5%
ayr	1	100.0%		0.0%
b	849	100.0%		0.0%
c	443	98.9%	5	1.1%
ca	232	95.5%	11	4.5%
ce	131	96.3%	5	3.7%
ch	277	99.6%	1	0.4%
ci	71	94.7%	4	5.3%
ck	54	100.0%		0.0%

ตัวอักษร	สอดคล้อง		ไม่สอดคล้อง	
cl	96	98.0%	2	2.0%
co	250	97.7%	6	2.3%
cu	32	86.5%	5	13.5%
cy	37	100.0%		0.0%
cz	2	100.0%		0.0%
d	1182	99.0%	12	1.0%
e	1536	55.5%	1231	44.5%
ea	91	100.0%		0.0%
ear	6	100.0%		0.0%
eau	6	100.0%		0.0%
ee	60	100.0%		0.0%
eer	5	100.0%		0.0%
ei	42	100.0%		0.0%
eir	4	100.0%		0.0%
eo	46	86.8%	7	13.2%
eou	1	33.3%	2	66.7%
er	567	96.6%	20	3.4%
ere	10	90.9%	1	9.1%
eu	37	94.9%	2	5.1%
eur	3	100.0%		0.0%
ew	10	100.0%		0.0%
ey	84	95.5%	4	4.5%
f	394	99.7%	1	0.3%
g	76	95.0%	4	5.0%
ga	127	100.0%		0.0%
ge	130	95.6%	6	4.4%
gh	48	98.0%	1	2.0%
gi	58	98.3%	1	1.7%
gl	31	96.9%	1	3.1%
gn	5	100.0%		0.0%
go	93	100.0%		0.0%
gr	107	100.0%		0.0%
gu	48	100.0%		0.0%
gy	7	58.3%	5	41.7%

ตัวอักษร	สอดคล้อง		ไม่สอดคล้อง	
h	424	100.0%		0.0%
i	2365	99.7%	7	0.3%
ia	196	99.5%	1	0.5%
ie	85	90.4%	9	9.6%
ii	1	100.0%		0.0%
io		0.0%	6	100.0%
ion	45	81.8%	10	18.2%
ir	31	100.0%		0.0%
ire	21	95.5%	1	4.5%
iu	94	100.0%		0.0%
j	88	96.7%	3	3.3%
k	372	94.7%	21	5.3%
kh	6	100.0%		0.0%
kn	5	100.0%		0.0%
l	2338	99.7%	8	0.3%
m	1459	99.5%	8	0.5%
n	2605	100.0%	1	0.0%
nc	7	100.0%		0.0%
ng	93	100.0%		0.0%
nk	2	100.0%		0.0%
o	2222	97.9%	48	2.1%
oa	32	100.0%		0.0%
oe	28	82.4%	6	17.6%
oi	62	100.0%		0.0%
oo	59	100.0%		0.0%
oor	5	100.0%		0.0%
or	248	99.6%	1	0.4%
ore	21	100.0%		0.0%
ou	92	100.0%		0.0%
oui		0.0%	5	100.0%
our	12	100.0%		0.0%
ow	65	97.0%	2	3.0%
oy	21	100.0%		0.0%
p	703	87.4%	101	12.6%

ตัวอักษร	สอดคล้อง		ไม่สอดคล้อง	
ph	157	100.0%		0.0%
pn	1	100.0%		0.0%
ps	6	100.0%		0.0%
pt	7	100.0%		0.0%
q	3	100.0%		0.0%
qu	76	95.0%	4	5.0%
r	2941	99.1%	26	0.9%
rh	26	100.0%		0.0%
s	1805	99.9%	1	0.1%
sc	3	75.0%	1	25.0%
sch	10	100.0%		0.0%
sh	94	97.9%	2	2.1%
sk	25	100.0%		0.0%
sm	18	85.7%	3	14.3%
sp	49	100.0%		0.0%
st	68	100.0%		0.0%
t	2074	91.7%	187	8.3%
tb		0.0%	1	100.0%
th	185	99.5%	1	0.5%
thm	3	100.0%		0.0%
ti	48	98.0%	1	2.0%
ts		0.0%	1	100.0%
u	662	99.7%	2	0.3%
ua	22	100.0%		0.0%
ue	8	80.0%	2	20.0%
ui	13	81.3%	3	18.8%
uo		0.0%	1	100.0%
uoi		0.0%	1	100.0%
ur	55	93.2%	4	6.8%
ure	7	100.0%		0.0%
uy	3	75.0%	1	25.0%
v	322	99.7%	1	0.3%
w	336	96.0%	14	4.0%
wh	10	100.0%		0.0%

ตัวอักษร	สอดคล้อง		ไม่สอดคล้อง	
wr	1	100.0%		0.0%
x	108	98.2%	2	1.8%
y	675	96.6%	24	3.4%
ya		0.0%	2	100.0%
ye	2	33.3%	4	66.7%
yo		0.0%	1	100.0%
yr	1	100.0%		0.0%
yu		0.0%	3	100.0%
z	164	98.2%	3	1.8%
รวม	34580	94.7%	1936	5.3%

ตาราง 19: ผลการเปรียบเทียบความสอดคล้องของข้อมูลจริงกับหลักเกณฑ์การทับศัพท์

จากตาราง จะเห็นว่า เมื่อพิจารณาภาพรวม แม้เป็นข้อมูลทับศัพท์จากราชบัณฑิตยสถานเอง ความสอดคล้องโดยรวมมีเพียง 95% ทั้งนี้เป็นเพราะเกณฑ์ที่ทางราชบัณฑิตยสถานได้วางไว้เป็นเกณฑ์สำหรับให้มนุษย์ใช้จึงไม่ได้ครอบคลุมรายละเอียดที่ปลีกย่อยมากๆ แต่เมื่อให้คอมพิวเตอร์แยกแยะข้อมูลของคำทับศัพท์ที่พบจริง จึงเห็นว่ามีตัวอักษรจำนวนหนึ่งที่ไม่ได้ถอดอักษรในลักษณะเดียวกับที่ระบุไว้ในเกณฑ์ หรือเกณฑ์ไม่ได้กำหนดไว้ชัดเจนพอ ตัวอักษรอื่นๆที่พบว่าถอดอักษรมาไม่สอดคล้องกับเกณฑ์และพบเป็นจำนวนพอสมควร ได้แก่

- ตัวอักษรที่ดูเหมือนเป็นปัญหามากที่สุดคือ ตัวอักษร e ซึ่งเมื่อถอดอักษรแล้วไม่เป็นไปตามเกณฑ์ถึง 1231 ครั้งหรือ 44% แต่แท้จริงแล้วเป็นตัวอักษร e ท้ายคำที่ไม่ออกเสียงถึง 1162 ครั้ง เช่น scale - สเกล, serve - เสิร์ฟ, grade - เกรด, carbide - คาร์ไบด์ มีเพียงจำนวนหนึ่งเท่านั้นที่ถอดต่างไปจากเกณฑ์ เช่น ถอดเป็น เออ (nikel - นิเกิล, cable - เคเบิล) ถอดเป็น แอ (lumen - ลูแมน, spitbergen - สปิตสเบอร์แกน)
- ตัวอักษร eo นอกหนือจากการถอดเป็น อี เอ เอีย เอียว แล้วยังพบว่าถอดเป็น ออ เช่น george - จอร์จ, georgia - จอร์เจีย
- ตัวอักษร c ที่ปรากฏหน้า u หรือ o จำนวนหนึ่งพบว่าถอดเป็น ก ไม่ใช่ ค ตามที่ระบุไว้ในเกณฑ์ เช่น molecule - โมเลกุล, focus - โฟกัส, damuscus - ดามัสกัส, alcohol - แอลกอฮอล์, electroscope - อิเล็กโทรสโกป
- ตัวอักษร p ที่เป็นพยัญชนะต้นจำนวนหนึ่งถอดเป็น ป แทนที่จะเป็น พ ตามเกณฑ์ เช่น pamir - ปามีร์, trumpet - ทรัมเป็ต, petroleum - ปิโตรเลียม
- ตัวอักษร t ที่อยู่หน้า u ไม่ได้ถอด ท แต่ถอดเป็น จ เช่น amplitude - แอมพลิจูด, armature - อาร์มาเจอร์, venturi - เวนจูริ, ตัวอักษร t ในคำว่า vietname ถอดด้วย ด (vietnam-เวียดนาม) และมีคำจำนวนหนึ่งที่ถอด t เป็น ต แทนที่จะเป็น ท เช่น albertan - แอลเบอร์ตัน, brittany - บริตตานี, Pantar - ปันตาร์, hectare - เฮกตาร์, chesterfield - เชสเตอร์ฟิลด์, martin - มาร์ติน, metric - เมตริก, petroleum - ปิโตรเลียม เป็นต้น (แม้ว่าจะมีเกณฑ์ยกเว้นให้กับพยัญชนะบางกลุ่มที่คนไทยนิยมใช้

เสียง ต ได้แก่ anti-, auto-, inter-, multi-, photo-; ta, -ter, -ti, -tic, -ting, -tis, -to, -ton, - tor, -tre, - tum, -tus และ -ty โดยราชบัณฑิตยสถานได้ให้ตัวอย่าง antibody - แอนติบอดี, intercom - อินเตอร์คอม, computer - คอมพิวเตอร์, quantum - ควอนตัม, zygomata - ไซโกมาตา ข้อยกเว้นเหล่านี้ดูเสมือนให้ใช้กับพยางค์ต้นหรือพยางค์ท้ายของคำ ตัวอย่างที่ยกมาข้างต้น เช่น brittany จึงไม่เข้าเกณฑ์ยกเว้นนี้ หรือหากคิดว่า ข้อยกเว้นนี้ใช้กับพยางค์กลางคำได้ด้วย คำว่า battery - แบตเตอรี่ ก็จะสอดคล้องกับเกณฑ์ข้อยกเว้นให้ใช้ ต ได้ แต่ก็จะแย้งกับตัวอย่างอื่นๆ เช่น kettering - เคตเทอริง ซึ่งมี –ter- เช่นกัน แต่ถอดด้วย ท ไม่ใช่ ต โดยไม่เข้าข้อยกเว้นนี้)

- ตัวอักษร k ที่เป็นพยัญชนะต้นที่ปกติต้องถอดเป็น ค โดยมีข้อยกเว้นในกรณีที่ k เป็นพยัญชนะต้นของพยางค์สุดท้าย ให้ถอดเป็น ก ได้ เช่น bunker - บังเกอร์, market - มาร์เกต, Yankee - แยงกี จากข้อมูล พบว่ามีบางคำที่ k ในพยัญชนะต้นถอดอักษรเป็น ก แทนที่จะเป็น ค เช่น kilo - กิโล, kungar - กันการ์ และในพยางค์ท้ายบางคำก็ไม่ถอดด้วย ก ตามเกณฑ์ข้อยกเว้น แต่ถอดด้วย ค ตามเกณฑ์ปกติ เช่น algonkian - แอลกองเคียน, Makin - มาคิน, Bishkek - บิสเคค
- ตัวอักษร ay นอกจากถอดเป็น เอ ตามเกณฑ์แล้วยังพบว่าถอดเป็น ไอ หรือ อี ได้ เช่น panay - ปาไน, barcley - บาร์คล
- ตัวอักษร r บางกรณีไม่ออกเสียงจึงไม่ถอดเป็น ร เช่น Worcester - วูสเตอร์, divergent - ไดเวอเจนซ์, berners - เบอร์เนอส์ แต่ในคำปกติซึ่งตามเกณฑ์แม้ไม่ออกเสียงก็ถอดเป็น ร แล้วใส่เครื่องหมายทัณฑฆาตกำกับไว้ เช่น egbert - เอกเบิร์ต, barley - บาร์เลย์
- ในกรณีของ qu ท้ายพยางค์ซึ่งควรถอดเป็น ก พบว่ามีการถอดเป็น ค ในบางคำ เช่น technique - เทคนิค, cheque - เช็ค
- ตัวอักษร w ในบางคำพบว่าไม่ออกเสียงจึงไม่ถอดเป็นอักษร ว ตามเกณฑ์ เช่น woolwich - วูลิช, greenwich - กรีนิช, bungalow - บังกะโล ในขณะที่ส่วนมากเมื่อไม่ออกเสียงจะถอดเป็น ว และกำกับด้วยเครื่องหมายทัณฑฆาต เช่น snow - สโนว์, fellow - เฟลโลว์, lewis - ลูว์อิส
- ตัวอักษร y บางครั้งก็ไม่ออกเสียงและไม่ถอดเป็นอักษรใดใด เช่น pepys - พีปส์, heyer - เฮเออร์ ซึ่งในคำลักษณะเดียวกันมีการออกเสียงและถอดเป็น ย เช่น untermeyerซอุนเทอร์ไมเยอร์ หรือในบางกรณีก็พบว่าถอดทั้งสองแบบ เช่น Jersey - เจอร์ซี และ Jersey - เจอร์ซีย์

สาเหตุของความไม่สอดคล้องกับเกณฑ์ที่วางไว้นั้น เป็นไปได้ในหลายกรณี เช่น มีข้อยกเว้นว่าถ้าเป็นคำทับศัพท์ที่ใช้กันมานานจนถือเป็นภาษาไทย และปรากฏในพจนานุกรม ฉบับราชบัณฑิตยสถานแล้ว ให้ใช้ต่อไปตามเดิม เช่น ช็อกโกเลต, ช็อกโกแลต, เชิ้ต, ก๊าซ, แก๊ส หรือมีข้อยกเว้นให้คำวิสามานยนามที่ใช้กันมานานแล้ว อาจใช้ต่อไปตามเดิมได้ เช่น Victoria - วิกตอเรีย, Louis - หลุยส์, Cologne - โคโลญ หรือคำวิสามานยนาม ที่ในภาษาอังกฤษออกเสียงเฉพาะพิเศษนอกเหนือจากที่กำหนดไว้ในตารางเกณฑ์การถอดอักษร ก็ให้ถอดตามการออกเสียง เช่น Worcester - วูสเตอร์, Marble Arch - มาร์บะลาช นอกจากนี้ คำบางคำที่เพิ่งบัญญัติใหม่แต่ไม่ตรงตามเกณฑ์ก็ต้องยึดเอาตามที่ปรากฏในพจนานุกรมหรือในหนังสือศัพท์บัญญัติ เช่น internet - อินเทอร์เน็ต ไม่ใช่ อินเตอร์เน็ต เพราะแม้ว่าจะเข้ากฎข้อยกเว้นของกลุ่มพยัญชนะบางกลุ่มที่ไทยเรานิยมใช้เสียง ต (anti-, auto-, inter-, multi-, photo-; ta, -ter, -ti, -tic, -ting, -tis, -to, -ton, - tor, -tre, - tum, -tus และ -ty) แต่เนื่องจากเป็นศัพท์บัญญัติทางคอมพิวเตอร์โดยใช้ ท ก็ต้องยึดตามหนังสือ

ด้วยเหตุต่างๆ นี้ คำบางคำเมื่อพิจารณาแล้วจึงอาจไม่ตรงตามเกณฑ์ได้ คำหล่านี้จึงเป็นปัญหาแม้กระทั่งกับมนุษย์เองในการถอดอักษร หากไม่รู้มาก่อนว่าได้มีการทับศัพท์มาก่อนที่แตกต่างจากเกณฑ์ ดังนั้น ในการประเมินประสิทธิภาพของโปรแกรมถอดอักษรภาษาอังกฤษเป็นไทย จึงต้องนำข้อจำกัดเหล่านี้มาพิจารณาด้วย กล่าวคือ เราคงไม่สามารถประเมินผลโดยเทียบผลลัพธ์ที่ได้จากโปรแกรมว่าตรงกับที่ปรากฏในข้อมูลได้เสมอ จำเป็นต้องนำคำที่ถอดอักษรแล้วไม่ตรงกับข้อมูลที่พบมาพิจารณาด้วยว่า โปรแกรมได้ถอดอักษรถูกต้องตามเกณฑ์แต่บังเอิญเป็นคำที่ต้องยกเว้นหรือไม่ ซึ่งจะได้กล่าวถึงในหัวข้อการวิเคราะห์ปัญหาของการถอดอักษรโดยอัตโนมัติด้วยคอมพิวเตอร์

3.3.3 การประเมินความถูกต้องของการถอดอักษรโดยอัตโนมัติด้วยคอมพิวเตอร์

เพื่อที่จะเข้าใจถึงปัญหาและวางแนวทางในการพัฒนาโปรแกรมถอดอักษรให้ทำงานได้เต็มประสิทธิภาพ จึงจำเป็นต้องวิเคราะห์ข้อผิดที่พบว่า การที่โปรแกรมถอดอักษรแล้วไม่ตรงกับคำไทยที่เขียนในข้อมูลนั้น เป็นข้อผิดจริงๆหรือเป็นเพียงการไม่ตรงกันด้วยสาเหตุของความหลากหลายในการออกเสียงหรือเป็นเพราะคำนั้นเป็นคำทับศัพท์แต่เดิมซึ่งอาจไม่ตรงตามเกณฑ์ตารางเทียบพยัญชนะและสระ ทั้งนี้เพราะโปรแกรมนี้ควรจะใช้เฉพาะกับการทับศัพท์คำใหม่ๆ ซึ่งผู้ใช้ต้องการให้ตรงตามเกณฑ์ที่ระบุในตารางการเทียบอักษรของราชบัณฑิตยสถานมากที่สุด แต่หากเป็นคำเก่าซึ่งทับศัพท์มานานแล้วและไม่ตรงตามเกณฑ์การเทียบอักษร คำเหล่านี้ควรรวมไว้ในหมวดของคำยกเว้น ซึ่งโปรแกรมควรจำรายการคำเหล่านี้ไว้และตรวจสอบข้อมูลเข้าว่าเป็นคำเหล่านี้หรือไม่ หากใช่ก็แสดงคำทับศัพท์ของคำนั้นออกมา แต่หากเป็นคำอื่นๆที่ไม่เข้าข้อยกเว้น จึงใช้ตารางกฎที่มีสร้างคำทับศัพท์ขึ้นมา

ผลการวิเคราะห์รายการคำที่ถอดอักษรไม่ตรงกับข้อมูลที่มี โดยวิเคราะห์จากผลการทดลองระบบแบบตารางกฎที่ใช้ทั้งรูปและเสียงอ่านภาษาไทยกับข้อมูลชุดแรก ซึ่งโปรแกรมถอดอักษรคำไม่ตรงกับในข้อมูลที่ใช้ทดสอบเป็นจำนวน 562 คำหรือคิดเป็น 34.4% จากการพิจารณารายคำทั้ง 562 คำนี้ พบว่ามีคำจำนวน 209 คำที่ถือได้ว่าเป็นการทับศัพท์ที่ยอมรับได้ตามเกณฑ์ มี 88 คำที่ไม่ถูกต้องแต่สามารถปรับปรุงโดยการเพิ่มเติมกฎเพื่อปรับให้ถูกต้องได้ มี 265 คำที่ถือได้ว่าเป็นเป็นการถอดอักษรที่ผิด โดยในจำนวนนี้มี 41 คำที่ได้คำที่ไม่ถูกตามอักขรวิธีภาษาไทย ดังมีรายละเอียดดังนี้

คำที่โปรแกรมถอดอักษรแล้วต่างจากข้อมูลที่พบ แต่ก็ยังถือว่ายอมรับได้ตามเกณฑ์ ตัวอย่างเช่น olympus โปรแกรมถอดเป็น โอลิมพัส แต่ข้อมูลที่พบเป็น โอลิมปัส กรณีนี้จะไม่ถือว่าโปรแกรมถอดอักษรผิด เพราะตามหลักการเทียบพยัญชนะ p ที่เป็นพยัญชนะต้นจะถอดเป็น พ แต่ที่คำนี้ลงท้ายด้วย -pus ซึ่งคนไทยนิยมใช้เสียง พ มากกว่า สิ่งที่ควรทำเพิ่มเติมคือการเพิ่มโมดูลเพื่อตรวจแก้คำเหล่านี้ให้เข้ากับข้อยกเว้นนี้. palmerston โปรแกรมถอดเป็น พาลเมอร์สตัน แต่ข้อมูลคำทับศัพท์ที่พบเป็น พัลเมอร์สตัน กรณีนี้ก็ถือว่าพอยอมรับได้ เพราะเป็นความต่างที่ความสั้นยาวของเสียงสระว่าเป็น อะ หรือ อา ซึ่งแต่ละคนก็อาจออกเสียงสั้นยาวนี้ต่างกันได้. ballast โปรแกรมถอดเป็น บัลลาสต์ ในขณะที่คำทับศัพท์เป็น แบลลาสต์ กรณีนี้ก็ถือได้ว่าเป็นความต่างของสำเนียงที่ใช้อ่าน จึงน่าจะยอมรับได้

คำที่ถอดอักษรมาไม่ถูกต้องตามเกณฑ์ แต่เมื่อพิจารณาดูแล้วเป็นข้อผิดเล็กน้อย ซึ่งสามารถเพิ่มโมดูลเพื่อปรับให้ตรงตามเกณฑ์ได้ ตัวอย่างเช่น คำว่า cyclic ซึ่งโปรแกรมถอดเป็น ไซกลิก แต่คำทับศัพท์เป็น ไซคลิก. คำว่า ikaria โปรแกรมถอดอักษรเป็น อิกาเรีย แต่คำทับศัพท์เป็น อิคาเรีย. คำว่า diploid โปรแกรมถอดเป็น ดิปลอยด์ แต่คำทับศัพท์เป็น ดิพลอยด์ กรณีเหล่านี้สามารถแก้ไขได้โดยการตรวจสอบพยัญชนะต้น

c, k, p ว่าควรเป็น ค, ค, พ ตามลำดับ ในบางคำโปรแกรมถอดอักษรโดยใส่เครื่องหมายทัณฑฆาตเกินมาหรือใส่ผิดตำแหน่ง เช่น andrews โปรแกรมถอดเป็น แอนดรูว์ส แทนที่จะเป็น แอนดรูวส์ หรือคำว่า amps โปรแกรมถอดมาเป็น แอมป์ส์ แทนที่จะเป็น แอมปส์ ในบางคำ โปรแกรมถอดอักษรคำควบกล้ำผิดไป เช่น electrolytic ถูกถอดเป็น อิเล็กทโรลิติก, clyde ถูกถอดเป็น คไลด์ เป็นต้น กรณีเหล่านี้ น่าจะปรับแก้ไขโดยกฎที่เขียนขึ้นเพื่อตรวจตามอักขรวิธีภาษาไทยได้

นอกจากนั้น จะเป็นคำซึ่งถือได้ว่าโปรแกรมถอดอักษรไม่ถูกต้อง ซึ่งมีจำนวน 280 คำ ในจำนวนนี้มีคำอยู่ 41 คำที่ได้คำที่ไม่ถูกต้องตามอักขรวิธีภาษาไทย ดังนั้น เมื่อประเมินประสิทธิภาพในการถอดอักษรได้ถูกต้องของโปรแกรมในการทดลองแรกจริงๆ จึงไม่ใช่ 64.8% แต่เป็น 77.4% (1060 + 207) โดยที่ยังมีคำอีกจำจนวนหนึ่ง (89 คำ) ซึ่งน่าจะแก้ไขด้วยการเพิ่มเติมโมดูลตรวจสอบและปรับผลให้ถูกต้องได้ ดังนั้น จึงอาจกล่าวได้ว่า ผลของการถอดอักษรอังกฤษเป็นไทยในคำที่ไม่เคยเห็นมาก่อนนั้นในการทดลองครั้งนี้อยู่ที่ 65-77%

ตรงกับข้อมูล	64.8%	1060			1267	77.4%	ถูก
ไม่ตรงกับข้อมูล	36.2%	576	ยอมรับได้	207			
			ปรับแก้ไขได้	89			
			ผิดพลาด	280	369	21.6%	ผิด

ตาราง 20: ผลการนับค่าความถูกต้องใหม่ (ข้อมูลชุด 1 ระบบแบบตารางกฎที่ใช้ทั้งรูปและเสียงอ่านภาษาไทย)

และเมื่อนำผลจากการทดลองใช้ระบบเอ็นแกรมแบบกลุ่มที่ใช้เสียงอ่านภาษาไทยในการทดลองกับข้อมูลชุดแรก ซึ่งได้ผลที่ไม่ตรงกับข้อมูลที่มีจำนวน 530 คำหรือ 32.4% มาวิเคราะห์ในลักษณะเดียวกัน ก็พบว่า ในจำนวนคำ 530 คำที่ได้ผลไม่ตรงกับข้อมูลนั้น มีคำจำนวน 273 ที่ถือว่ายอมรับได้ มี 43 คำที่ได้ผลไม่ถูกต้องแต่น่าจะแก้ไขด้วยการเพิ่มโมดูลเพื่อแก้ไขตัวอักษร และมี 214 คำที่ผลการถอดอักษรยอมรับไม่ได้ ดังนั้น ความถูกต้องของระบบที่ได้จากการทดลองครั้งนี้จึงอยู่ที่ 67.6%-84.3%

ตรงกับข้อมูล	67.6%	1106			1379	84.3%	ถูก
ไม่ตรงกับข้อมูล	32.4%	530	ยอมรับได้	273			
			ปรับแก้ไขได้	43			
			ผิดพลาด	214	257	16.7%	ผิด

ตาราง 21: ผลการนับค่าความถูกต้องใหม่ (ข้อมูลชุด 1 ระบบแบบเอ็นแกรมแบบกลุ่มที่ใช้เสียงอ่านภาษาไทย)

3.4 สรุปผลและข้อเสนอแนะ

ด้วยข้อจำกัดของความถูกต้องของผลที่ได้ซึ่งไม่สูงมากพอ จึงยังไม่เหมาะสมที่จะเผยแพร่โปรแกรมที่พัฒนานี้เพื่อใช้เป็นเครื่องมือการเขียนคำทับศัพท์สำหรับผู้ใช้ทั่วไป จำเป็นต้องศึกษาและพัฒนาโปรแกรมให้มีประสิทธิภาพให้ดีมากขึ้น ซึ่งจากผลการวิเคราะห์ข้อผิดที่กล่าวมาแล้ว พบว่าคำทับศัพท์ส่วนหนึ่งเป็นคำที่ใช้กันมาก่อนและถอดอักษรไม่ตรงตามตารางเทียบพยัญชนะและสระภาษาอังกฤษ คำเหล่านี้มีจำนวนจำกัด แนวทางหนึ่งที่ทำได้ คือให้จัดเก็บคำเหล่านี้เป็นรายการคำยกเว้น ซึ่งโปรแกรมไม่ควรทำการถอดอักษร แต่ให้จำคำทับศัพท์ที่ควรเป็นเลย ดังนั้น เมื่อโปรแกรมรับข้อมูลมา ให้ตรวจสอบก่อนว่าหากไม่ใช่คำที่อยู่ในรายการยกเว้น ก็ให้ถอดอักษรเป็นภาษาไทยได้ ซึ่ง ณ ปัจจุบันนี้ วิธีการที่ให้ผลถูกต้องมากสุดคือการถอดอักษรแบบจับคู่อักษรทั้งสองภาษาพร้อมทั้งใช้เสียงอ่านแบบเป็นกลุ่มก้อน แต่ข้อเสียของวิธีการนี้ คือใช้หน่วยความจำ

ของคอมพิวเตอร์มากและประมวลผลช้า จึงอาจไม่เหมาะต่อการใช้งานจริง ซึ่งต้องการความรวดเร็วมากกว่านี้ วิธีการที่ให้ผลดีรองลงมาคือระบบที่ใช้ตารางกฎที่ใช้ทั้งรูปอักษรและหน่วยเสียง ซึ่งแม้จะให้ค่าความถูกต้องที่น้อยกว่าแต่สามารถประมวลผลได้เร็วกว่า หลังจากที่ได้รูปคำทับศัพท์จากโปรแกรมแล้วจึงเข้าโมดูลเพื่อปรับแต่งคำให้ถูกต้องมากยิ่งขึ้น ซึ่งในโมดูลนี้จะตรวจสอบการถอดอักษรพยัญชนะต้นและพยัญชนะสะกด พ-ป, ก-ค, ต-ท ตรวจสอบการใส่เครื่องหมายทัณฑฆาต และปรับปัญหาคำควบกล้ำที่กล่าวมาแล้ว จึงควรมีการพัฒนาโมดูลนี้เพื่อวิเคราะห์ว่าสามารถทำให้ได้ผลดีขึ้นอย่างที่คาดหรือไม่ และส่งผลกระทบต่อคำที่โปรแกรมเคยถอดอักษรไว้ถูกต้องแล้วหรือไม่

ข้อจำกัดอีกข้อของโปรแกรมถอดอักษรภาษาอังกฤษเป็นภาษาไทยซึ่งนับเป็นข้อจำกัดอย่างมากในขณะนี้ คือความสามารถของโมดูลถ่ายเสียงภาษาอังกฤษ ด้วยระดับความถูกต้องของโมดูลถ่ายเสียงภาษาอังกฤษที่ 57% ในระดับคำ และ 90% ในระดับหน่วยเสียง การที่โปรแกรมถอดอักษรที่พัฒนาสามารถถอดอักษรอังกฤษเป็นไทยได้ถูกต้อง 65-84% ในระดับคำ จึงนับว่าสูงมากแล้ว หากไม่สามารถพัฒนาโมดูลถ่ายเสียงภาษาอังกฤษที่มีประสิทธิภาพมากกว่าที่เป็นอยู่ได้ ก็ยากที่จะพัฒนาโปรแกรมถอดอักษรภาษาอังกฤษเป็นไทยได้มากกว่าที่เป็นอยู่ แนวทางการพัฒนาต่อไปที่ควรทำ จึงเป็นเรื่องของการพัฒนาโมดูลถ่ายเสียงภาษาอังกฤษให้ถูกต้องมากขึ้น แต่ปัญหาการถ่ายเสียงคำภาษาอังกฤษเป็นงานที่ยาก หากต้องการให้ได้คำอ่านที่ถูกต้องจริง แนวทางที่งานวิจัยด้านการสังเคราะห์เสียงภาษาอังกฤษนิยมทำกัน คือ การเก็บคำอ่านไว้ในพจนานุกรมภาษาอังกฤษที่ใช้เลย กฎการถ่ายเสียงจะใช้เฉพาะกรณีกับคำซึ่งไม่อยู่ในพจนานุกรม เช่น ระบบสังเคราะห์เสียงของ AT&T (อ้างใน Llitjós 2001) ส่วนการสร้างกฎการอ่านออกเสียงนั้น ในปัจจุบัน นิยมใช้วิธีการสร้างโดยอัตโนมัติมากกว่าการสร้างเองด้วยนักภาษาศาสตร์ ทั้งนี้เพราะกฎที่เครื่องสร้างมักมีประสิทธิภาพสูงกว่า (Chotimongkol and Black, 2001) แต่ถึงอย่างไรก็ตาม ค่าความถูกต้องในระดับคำของเสียงอ่านที่ได้ก็ไม่สูงมากนัก Black et al. (1998) รายงานความถูกต้องของเสียงอ่านที่ได้จากการใช้ข้อมูล CMUDICT (ข้อมูลเดียวกับที่ใช้ในงานวิจัยนี้) เพียง 57.80% ในระดับคำ และ 91.99% ในระดับหน่วยเสียง[1] ดังนั้น โมดูลถ่ายเสียงภาษาอังกฤษที่พัฒนาขึ้นใช้ในงานวิจัยนี้จึงให้ผลที่ใกล้เคียงกับงานวิจัยอื่น (56.7% ในระดับคำ และ 90.5% ในระดับหน่วยเสียง)

นอกจากนี้ ในงานถอดอักษรภาษาอังกฤษเป็นไทยนี้ คำจำนวนมากที่ผู้ใช้ต้องการทับศัพท์มักจะเป็นชื่อเฉพาะ ซึ่งการถ่ายเสียงชื่อเฉพาะถือเป็นงานที่ยากกว่าการถ่ายเสียงคำทั่วไป Llitjós (2001) กล่าวสรุปถึงปัญหาของการถ่ายเสียงอ่านชื่อเฉพาะว่าเกิดจากสาเหตุต่างๆ ได้แก่ (1) การที่ชื่อจำนวนมาก มักมีที่มาจากรากภาษาแตกต่างกันซึ่งนำเข้ามาในอีกภาษาโดยไม่ผ่านกระบวนการกลายเสียงในภาษานั้น (assimilation) ทำให้สรุปเป็นหลักการอ่านออกเสียงได้ยาก (2) จำนวนของชื่อเฉพาะมีปริมาณมาก เช่น มีผู้รวบรวมชื่อสกุลภาษาอังกฤษได้ไม่ต่ำกว่า 1.5 ล้านชื่อที่แตกต่างกันจากข้อมูลประชากร 7 ล้านคน จึงเป็นเรื่องค่อนข้างยากที่จะเก็บข้อมูลชื่อเฉพาะให้ครบถ้วนไว้ในพจนานุกรมคำอ่านได้หมด (3) ลักษณะการอ่านออกเสียงชื่อเฉพาะอาจเป็นลักษณะเฉพาะตัวที่ไม่ตรงตามกฎการอ่านออกเสียงในภาษาก็ได้ การถ่ายเสียงภาษาอังกฤษเพื่อการทับศัพท์ภาษาอังกฤษในงานวิจัยนี้จึงเป็นปัญหาใหญ่ที่อาจส่งผลให้ระบบทำงานได้ไม่ดีเท่าที่ต้องการ

---

[1] ในขณะที่ความถูกต้องเมื่อทดสอบกับข้อมูลพจนานุกรมอื่น เช่น Oxford Advanced Learners Dictionary of Contemporary English จะให้ความถูกต้องที่สูงกว่าถึง 74.56% Black et al. ให้เหตุผลว่าเพราะข้อมูลใน CMUDICT นั้นมีชื่อเฉพาะอยู่เป็นจำนวนมาก)

แต่เมื่อกลับไปพิจารณาโปรแกรมถอดอักษรไทยเป็นโรมันซึ่งได้ผลความถูกต้องสูงกว่ามากคือได้ 94.44% สำหรับชื่อเฉพาะไทย และ 99.58% สำหรับคำทั่วไป คำถามที่ตามคือความแตกต่างของความถูกต้องในระบบการถอดอักษรจากไทยเป็นอังกฤษ และจากอังกฤษเป็นไทย เกิดจากความยากของภาษาอังกฤษเอง หรือเกิดจากสาเหตุอื่น หากพิจารณาผลความถูกต้องของการถ่ายเสียงภาษาไทยที่ Chotimongkol and Black (2002) รายงานไว้ที่ 68.76% โดยการสร้างกฎการถ่ายอักษรไทยเป็นเสียง เทียบกับความถูกต้องของระบบการถอดอักษรไทยเป็นโรมันที่พัฒนาขึ้นในงานวิจัยนี้โดยการแก้ปัญหาความกำกวมในการอ่านในระดับพยางค์และคำไม่ใช่ระดับอักขระ ก็อาจเป็นไปได้ว่า ยังมีหนทางอื่นที่เหมาะสมกว่าในการถ่ายเสียงภาษาอังกฤษเพื่อใช้ในการทับศัพท์ ความถูกต้องที่ต่ำนั้นอาจเกิดจากการเลือกระดับของการวิเคราะห์ปัญหาที่ไม่เหมาะสมกับปัญหานี้ จึงเป็นประเด็นปัญหาที่ต้องวิจัยต่อเนื่องอีก

บรรณานุกรม

ALA-LC 1997. *Romanization tables : transliteration schemes for non-roman scripts 1997*, approved by the Library of Congress and the American Libraty Association, Randall K. Barry (editor). Washing-ton, D.C.: Library of Congress, 1997 (http://www.loc.gov/catdir/cpso/roman.html)

Al-Onaizan, Y. and K. Knight. 2002. Machine Transliteration of Names in Arabic Text. In *Proceedings of ACL Workshop on Computational Approaches to Semitic Languages*. pages 34-46, Philadelphia, USA.

Aroonmanakun, W. 2002. Collocation and Thai Word Segmentation. In *Proceedings of the Fifth Symposium on Natural Language Processing & The Fifth Oriental COCOSDA Workshop*, pages 68-75, Pathumthani: Sirindhorn International Institute of Technology.

Aroonmanakun, W., W. Rivepiboon. 2004. A Unified Model of Thai Word Segmentation and Romanization. In *Proceedings of The 18th Pacific Asia Conference on Language, Information and Computation*, Dec 8-10, 2004, Tokyo, Japan. 205-214.

Black, A. W., Lenzo, K. and Pagel, V. 1998. Issues in Building General Letter to Sound Rules. *3rd ESCA Speech Synthesis Workshop*, Jenolan Caves, Australia, 77-80.

Bosch, A. van den and W. Daelemans. 1993. Data-oriented methods for grapheme-to-phoneme conversion. In *Proceedings of the Sixth Conference of the European Chapter of the Association for Computational Linguistics*, Utrecht, Netherland., 45-53,

Charoenporn, Thatsanee, Ananlada Chotimongkol, and Virach Sornlertlamvanich. 1999. Automatic Romanization for Thai. *Proceedings of the Second International Workshop on East-Asian Language Resources and Evaluation* (ORIENTAL COCOSDA '99), 137-140. (ftp://www.links.nectec.or.th/pub/paper/virach/cocosda99.ps.gz)

Chen, Hsin-Hsi, Sheng-Jie Huang, Yung-Wei Ding, and Shih-Chung Tsai. 1998. Proper Name Translation in Cross-Language Information Retrieval. In *Proceedings of the 36th Annual Meeting of the Association for Computational Linguistics and 17th International Conference on Computational Linguistics*, Montreal, Quebec, Canada, 232-236,

Chen, Stanley F., and Joshua Goodman. 1998. *An Empirical Study of Smoothing techniques for Language Modeling*. TR-10-98. Harvard University.

Chotimongkol, Ananlada and Alan W Black. 2000. Statistically trained orthographic to sound Models for Thai, In *Proceedings of ICSLP 2000*, Beijing, China October 2000.

Fujii, Atsushi and Tetsuya Ishikawa. 1999. Cross-Language Information Retrieval for Technical Documents. In *Proceedings of the Joint ACL SIGDAT Conference on Empirical Methods in Natural Language Processing and Very Large Corpora*, pages 29-37,

Fujii, Atsushi and Tetsuya Ishikawa. 2001. Japanese/English Cross-Language Information Retrieval: Exploration of Query Translation and Transliteration. In *Computers and the Humanities*, 35(4): 389-420. (http://arXiv.org/abs/cs/0206015)

Glover, Bonnie and Kevin Knight. 1998. Translating names and technical terms in arabic text. In *Proceedings of the COLING/ACL Workshop on Computational Approaches to Semitic Languages*. Montreal, Quebec, Canada, 34-41.

Goto, Isao, Naoto Kato, Noriyoshi Uratani, and Terumasa Ehara. 2003. Transliteration Considering Context Information based on the Maximum Entropy Method. In *Proceedings of Machine Translation Summit IX*, New Orleans, Louisiana, USA.

Haizhou, Li, Zhang Min, and Su Jian. 2004. A Joint Source-Channel Model for Machine Transliteration. In *Proceedings of the 42nd Annual Meeting of the Association for Computational Linguistics*, Barcelona, Spain, 159-166.

ISO/FDIS 11940. 1998. Information and Documentation – Transliteration of Thai.

Jung, SungYoung, SungLim Hong, and Eunok Paek. 2000. An English to Korean Transliteration Model of Extended Markov Window. In *Proceedings of the 18th International Conference on Computational Linguistics*, Saarbrücken, Germany, 383-389.

Kang, In-Ho and GilChang Kim. 2000. English-to-Korean Transliteration using Multiple Unbounded Overlapping Phoneme Chunks. In *Proceedings of the 18th International Conference on Computational Linguistics*, Saarbrücken, Germany, 418-424.

Kawtrakul, Asanee, Amarin Deemagarn, Chalathip Thumkanon, Navapat Khantonthong, and Paul McFetridge. 1998. Backward Transliteration for Thai Document Retrieval. *IEEE Asia Pacific Conference on Circuits and Systems*, 1998.

Kawtrakul, A. Thumkanon, C., Poovorawan, Y., and Suktarachan, M. 1997. Automatic Thai Unknown Word Recognition. In Proceedings of the natural language Processing Pacific Rim Symposium 1997 (NLPRS'1997).

Khamya, A., L. Narupiyakul, and B. Sirinaovakul, 2000. SATTS : Syllable Analysis for Text-To-Speech System, In *The 4th Symposium on Natural Language Processing (SNLP 2000)*, May 10-12, Chiang Mai Plaza Hotel, Chiang Mai, 336-340.

Knight, Kevin and Jonathan Graehl. 1997. Machine Transliteration. In *Proceedings of 35th Annual Meeting of the Association for Computational Linguistics and the 8th Conference of the European Chapter of the Association for Computational Linguistics*, Madrid, Spain, 128-135.

Kuo, Jin-Shea and Ying-Kuei Yang. 2004. Incorporating Pronunciation Variation into Different Strategies of Term Transliteration. In *Proceedings of The 18th Pacific Asia Conference on Language, Information and Computation*, Dec 8-10, 2004, Tokyo, Japan. 251-258.

Lee, Chun-Jen, Jason S. Chang, Jyh-Shing Roger JANG. 2003. A Statistical Approach to Chinese-to-English Back-Transliteration. In *Proceedings of 17th Pacific Asia Conference on Language, Information and Computation*, 1-3 October, 2003, Sentosa, Singapore.

Lenzo, Kevin A. (1998) "s/ ($text) / speech $1 / eg;" In *The Perl Journal*, Issue 12, pp.26-29.

Llitjós, Ariadna Font. 2001. *Improving Pronunciation Accuracy of Proper Names with Language Origin Classes*. Master Thesis. Carnegie Mellon University.

Londe, David L., Udom Warotamasikkhadit, and Nitaya Kanchanawan. 1971. *TRACTS . Thai-Roman Computerized Transliteration System*. Research Report sponsored by the Advanced Research Projects Agency for the Thai-US Military Research and Development Center.

Luksaneeyanawin, Sudaporn. 1990. A Thai Text to Speech System. In *Proceeding of the Conference of the Regional Workshops on Computer Processing of Asian Languages*, Asian Institute of Technology, 305-315.

Meknavin, Surapant and Boonserm Kijsirikul. 2000. Thai Grapheme-to-Phoneme Conversion. In Burnham, Denis, et.al. *Interdisciplinary Approaches to Language Processing: The International Conference on Human and Machine Processing of Language and Speech*. NECTEC: Bangkok, 215-224.

Meknavin, S., Charenpornsawat, P., and kijsirikul, B. 1997. Fetaure-based Thai Words Segmentation, *NLPRS, Incorporating SNLP-97*.

Poowarawan, Y. 1986. Dictionary-based Thai Syllable Separation. In *Proceeding of the Ninth Electronics Engineering Conference*.

Royal Institute. 1999. *Principles of Romanization for Thai Script by Transcription Method*

Suwanvisat, P. and S. Prasitjutrakul. 1998. Thai-English Cross-Language Transliterated Word Retrieval using Soundex Technique. In *The National Computer Science and Engineering Conference*, Kasetsart University, Bangkok, Thailand.

Suwanvisat, P. and S. Prasitjutrakul. 1999. Transliterated Word Encoding and Retrieval Algorithms for Thai-English Cross-Language Retrieval. In *The National Computer Science and Engineering Conference*, Bangkok, Thailand. (in Thai)

Tarsaku, Pongthai, Virach Sornlertlamvanich, and Rachod Thongpresirt. 2001. Thai Grapheme-to-Phoneme Using Probabilistic GLR Parser. In Proceedings of Eurospeech 2001, Aalborg, Denmark, Sept 2001.

Tesprasit, Virongrong, Paisarn Charoenpornsawat, and Virach Sornlertlamvanich. 2003. A Context-Sensitive Homograph Disambiguation in Thai Text-to-Speech Synthesis. In *Human Language Technology Conference 2003 (HLT-NAACL 2003)*, Edmonton, Canada, May 2003

Theeramunkong, T., Usanavasin, S., Machomsomboon, T., and Opasanont, B. 2000. Thai Word Segmentation without a Dictionary by Using Decision Trees. *The fourth Symposium on Natural Language Processing 2000.*

Wan, Stephen and Cornelia Maria Verspoor. 1998. Automatic English-Chinese name transliteration for development of multilingual resources. In *Proceedings of the 36th Annual Meeting of the Association for Computational Linguistics and 17th International Conference on Computational Linguistics*, Montreal, Quebec, Canada, 1352-1356

ผลที่ได้จากโครงการ

บทความวิจัย

1. Aroonmanakun, W., W. Rivepiboon. 2004. A Unified Model of Thai Word Segmentation and Romanization. In *Proceedings of The 18th Pacific Asia Conference on Language, Information and Computation*, Dec 8-10, 2004, Tokyo, Japan. 205-214.
2. A Chunk-based n-gram English to Thai Transliteration System. To be submitted to SNLP 2005 Conference.

โปรแกรม

1. โปรแกรมถอดอักษรไทยเป็นโรมัน สามารถดาว์นโหลดได้ที่ http://pioneer.chula.ac.th/~awirote/ling/download.html หรือ http://www.arts.chula.ac.th/~ling/tts/download.html

ภาคผนวก

ก. บทความ A Unified Model of Thai Word Segmentation and Romanization

ข. บทความ A Chunk-based n-gram English to Thai Transliteration System.

ค. การเผยแพร่และใช้ประโยชน์จากโปรแกรมถอดอักษรไทยเป็นโรมัน

PACLIC18

Proceedings of The 18th Pacific Asia Conference on Language, Information and Computation

December 8th - 10th, 2004
Waseda University, Tokyo, Japan

Edited by
Hiroshi Masuichi, Tomoko Ohkuma, Kiyoshi Ishikawa,
Yasunari Harada and Kei Yoshimoto

DATES: December 8 (Wednesday) - 10 (Friday), 2004
VENUE: International Conference Center, Waseda University, Tokyo

ORGANIZED BY
The Logico-Linguistic Society of Japan

SPONSORED BY
Waseda University
Media Network Center, Waseda University

SUPPORTED BY
Institute for DECODE, Waseda University
Language and Speech Science Research Laboratories, Waseda University
Corporate Research Group, Fuji Xerox Co., Ltd.

ACKNOWLEDGMENTS
Preparations for PACLIC18 were financially supported in part by Waseda University Grant for Special Research Projects: International Joint Research No. 2003C-201.
PACLIC18 is financially supported by Waseda University Grant for International Conference Operation.

Table of Contents

A Unified Model of Thai Romanization and Word Segmentation

Wirote AROONMANAKUN
Dept. of Linguistics
Chulalongkorn University
Bangkok 10330, Thailand
Wirote.A@chula.ac.th

Wanchai RIVEPIBOON
Dept. of Computer Engineering
Chulalongkorn University
Bangkok 10330, Thailand
Wanchai.R@chula.ac.th

Abstract

Thai romanization is the way to write Thai language using roman alphabets. It could be performed on the basis of orthographic form (transliteration) or pronunciation (transcription) or both. As a result, many systems of romanization are in use. The Royal Institute has established the standard by proposing the principle of romanization on the basis of transcription. To ensure the standard, a fully automatic Thai romanization system should be publicly made available. In this paper, we discuss the problems of Thai Romanization. We argue that automatic Thai romanization is difficult because the ambiguities of pronunciation are caused not only by the ambiguities of syllable segmentation, but also by the ambiguities of word segmentation. A model of automatic romanization then is designed and implemented on this ground. The problem of romanization and word segmentation are handled simultaneously. A syllable-segmented corpus and a corpus of word-pronunciation are used for training the system. The accuracy of the system is 94.44% for unseen names and 99.58% for general texts. When the training corpus includes some proper names, the accuracy of romanizing unseen names was increased from 94.44% to 97%. Our system performs well because it is designed to better suit the problem.

1 Introduction

The attempt to create a system of Romanization for Thai texts began since the 17th century by French missionaries (Griswold, 1960). However, the work was left unnoticed by other foreigners, who tend to romanize Thai words using their own languages notations. Issues of standardizing Thai romanization have been concerned again in the early 20th century (Frankfurt (1906), Petithuguenin (1912),Vajiravudh (1912,1931), Frankfurt et al. (1931)). Some systems of romanization are done on the basis of orthographic form, e.g. the system proposed by the ISO (ISO 11940 : 1998). Some systems are based on the pronunciation, such as the Royal Institute's system, Thiengburanathum's system (in his Thai-English dictionaries). Some are not totally based on orthographic form or pronunciation, e.g. the system proposed by King Rama VI.[1] This could explain why there are many ways to romanize a Thai word. For example, a common name like "เกรียงศักดิ์" can be romanized as "Kriangsak", "Kriengsakdi", "Kriengsak", or "Kreangsak". To lessen this problem, the Royal Institute has established the principle of romanization for Thai.[2] The principle has been endorsed by the United Nation (UN 2002) and slightly modified by the American Library Association and Library of Congress (ALA-LC 1997) for romanizing Thai scripts. Though the standard has been established, it is still not easy for general users to do romanization by hand. People tend to romanize Thai words on their own, rather than adhering to the principle. To help promoting the principle, Thatsanee et al. (1999) proposed an idea to develop an automatic romanization system. We agree with Thatsanee et al. that an automatic Thai romanization system is necessary for ensuring the standard. Such system was first developed in 1975 as a rule-based system (Londe et al. 1975). The accuracy was reported as greater than 95% when testing on Thai words. Unfortunately, the

[1] The system is known as a "graphic system" because it does not romanize words as pronounced in Thai. Words derived from Pali and Sanskrit words will be romanized to reflect the original words.

[2] The first version was proposed in 1939. The latest one is announced in 1999.

system runs on a mainframe and it is not available to the public. In addition, to be useful for the task, the automatic romanization system must be highly accurate, or near perfect. An accuracy of 70-90% on running text reported in many Thai text-to-speech systems is not adequate for this task. Therefore, we aim to develop a romanization system that is highly accurate, at least 99%.

2 Why Automatic Thai Romanization is Difficult?

Systems of Thai romanization that are totally based on the orthographic form like the transliteration system of ISO 1194 : 1998 is easy to be implemented because there is a one-to-one mapping from Thai to roman characters. But the system that is based on the pronunciation is more difficult because mapping from letters to sounds is not a one-to-one mapping. Letter-to-sound or grapheme-to-phoneme conversion systems are those used in text-to-speech applications. The difficulties on this task vary according to the characteristics of the language. The difficulties of transcribing Thai words are already discussed in many research papers, such as Luksaneeyanawin (1989), Meknavin and Kijsirikul. (2000), Khamya et al. (2000), Chotimongkol and Black (2000), Tarsaku et al. (2001), Tesprasit et al. (2003). In sum, Thai is an alphabetical language. There are 44 characters for 21 consonant sounds, 19 characters (including 3 consonant characters) for 24 vowel sounds (18 single vowels and 6 diphthongs), and 4 characters for tone markers (5 tones), and a number of characters for special symbols and numbers in Thai. Most of the following problems of Thai grapheme-to-phoneme conversion have been discussed in previous research. They are rearranged and clarified as follows:[3]

(a) Mapping of characters to sounds depends on the position of that character and its surrounding contexts. For example, the character "ก" is mapped to /kh/ when it is an initial consonant (e.g. "กา"-/khaa0/-{remain}[4]), but it is mapped to /k/ if it is a final consonant (e.g. "นาก"-/naak2/-{Naga}. The character "เ" usually maps to /e/, but if it is followed by a consonant and a vowel form "า", both vowel forms, "เ" and "า", will map to /aw/ (e.g. "เมา"-/maw0/-{drunk}).

(b) It is possible that a vowel sound may not be represented by any character at all. For example, the syllable "กด"-/kot1/-{press} consists of only two characters for initial and final consonants. The vowel sound /o/ in this syllable does not have any corresponding character. In the syllable "สวรรค์"-/san4/, even no vowel character is presented, the syllable is pronounced with a vowel /a/, as /san4/. But in "สวัสดิ์"-/sɔɔn4/, the vowel /ɔɔ/ is added.

(c) In some cases, it is ambiguous whether vowel forms belong to one syllable or two syllables. For example, in "เพลา", this string could be one syllable, "เพลา"-/phlaw0/-{axle}, in which "เ..า" represents the sound /aw/, or two syllables, "เพ- ลา"-/phee0-laa0/-{time}, in which "เ" and "า"represent different vowel sounds, /ee/ and /aa/ respectively.

(d) Since it is possible for two characters to represent one final consonant, it could be ambiguous whether the second character is a part of the final consonant, or it is the initial of the next syllable. For example, in "จักร", "ร" could be a part of the final consonants, "กร", as in "จักร-ยาน"-/cak1+ka1-jaan0/-{bicycle} or it could be the initial consonant of the following syllable, as in "จัก-รี"-/cak1-rii0/.

(e) Some characters map to different phonemes even they are in the same position (initial or final consonant). For example, character "ท", when used as an initial consonant, could map to either /d/ or /th/, as in "บัณ-ฑิต"-/ban0-dit1/-{graduate} and "มณ-ฑา"-/mon0-thaa0/-{Name of tree}.

(f) In some cases, one character can be both the final and initial consonants of two syllables. For example, the letter "ต" in "อัตรา"-{rate} represents the final consonant of the first syllable as well as the initial consonant of the following syllable, /?at1-traa0/.

---

[3] For each syllable, we will show only its orthographic form and its pronunciation. A gloss is not always provided because a syllable may or may not have a meaning in Thai. (A Thai word may composed of one or more syllables.) A gloss will be shown in { }.

[4] The numbers 0-4 at the end of syllable are used to represent five tones in Thai.

(g) There could be linking sounds between syllables in some words derived from Pali and Sanskrit. For example, in a compound word like "รัฐศาสตร์"-{political science} there is a linking syllable /tha1/ between the two words, "รัฐ"-/rat3/-{state} and "ศาสตร์"-/saat1/-{science}. This word is pronounced as /rat3+**tha1**-saat1/, rather than /rat3-saat1/.

(h) Even letters-to-sound conversion rules can be constructed, some syllables do not follow those rules. For example, the syllable "แม่น"-{precise} should be pronounced by rules with the long vowel as /mɛɛn2/, but it is actually pronounced with the short vowel as /mɛn2/. Though the vowel form "ไ" should map to a diphthong /aj/, but the syllable "ไห้" can be pronounced either as /haaj2/ (e.g. "ร้องไห้"-/rɔɔŋ3-haaj2/-{cry}) or /haj2/ (e.g. "เสาไห้"-/saw4-haj2/-{Name of rice}).[5]

(i) In some cases, a cluster of initial consonants can map to different phonemes. For example, "ปล" can map to a cluster sound /pl/, e.g. "ปลา"-/plaa0/-{fish}, or map to one leading syllable and an initial consonant sound /pa1-l/, e.g. "ปลาส"-/pa1+lat2/. The cluster "ทร" in a syllable can map to one phoneme /s/, e.g. "ทราบ"-/saap2/-{know}; or map to a cluster sound /thr/, e.g. "ทรา"-/thraa0/; or map to /tha3-r/, e.g. "ทรา"-/**tha3**-raa0/, depending on the word it occurs.

3 Model of Automatic Romanization

Since the Royal Institute's standard of romanization is based on pronunciation, it would be better to design the system to transcribe Thai texts first. The output from this system then will be useful not only to the romanization system but also to a Thai text-to-speech system. Besides, transforming the transcription output into roman characters is quite straightforward.

The process of transcription, or grapheme-to-phoneme conversion is a basic research in any languages. Many systems of Thai grapheme-to-phoneme have been proposed. Some are rule-based, such as Londe et al. (1975), Khamya et al. (2000). Some are dictionary and rule-based, such as Luksaneeyanawin (1989). Some are statistical based, such as Chotimongkol and Black (2000), Tarsaku et al. (2001). Some apply a machine learning method, such as Meknavin and Kijsirikul. (2000), Tesprasit et al. (2003). For other languages, statistical methods are commonly used in transcriptions and transliteration tasks, such as Bosch and Daelemans (1993), Knight and Graehl. (1997), Al-Onaizan and Knight (2002).

To design a system of grapheme-to-phoneme for Thai, besides the awareness of difficulties listed above, we have to choose the level of analysis that is right for the problems. In other words, what will be problems and solutions of grapheme-to-phoneme conversion are directly related to the design of the system. For example, for a string "เพลา", if the conversion is treated as a one-step process of mapping from character to sound, there will be a problem of determining whether "เ" should map to /e/ or should it be combined with "า" and mapped to /aw/. But if the conversion is treated as a multi-step process, in which syllable segmentation is the first step, the problem would be to determine whether the string "เพลา" is one syllable or two syllables. Furthermore, when dealing with linking syllables, if the system is designed to work at the character level, these linking syllables have to be generated from a character in a specific context. But if the system is designed to work at the syllable level, these linking syllables can be generated from a specific syllable in a certain context.

In our system[6], we prefer not to view grapheme-to-phoneme conversion in Thai as a one-step mapping from characters to sounds. We think that the process of syllabification is necessary. The input character strings will be converted to a sequence of syllables. During this process, all possible pronunciations of each syllable are generated. Then, the correct pronunciation and word boundaries will

[5] Although the problem of vowel length does not affect the result of romanization since the Royal Institute's romanization system does not differentiate between short and long vowels, we would like to take this problem into consideration since the system developed here is also used for a Thai text-to-speech system.

[6] The system is online and can be downloaded from http://www.arts.chula.ac.th/~ling/tts/

be determined. We share the same view with Meknavin and Kijsirikul. (2000), and Tesprasit et al. (2003) that pronunciation disambiguation should be done simultaneously with word segmentation. We think that it is not always possible to resolve pronunciation ambiguity by doing only syllable segmentation, as Thatsanee et al. (1999) suggested. For example, if "สักวา" is one word, "สักวา"-{a kind of poem}, there would be a linking syllable generated, /sak1+**ka1**-waa0/. But if these are two words, "สัก" -{about} and "วา"-{unit of measurement}, it would be pronounced without a linking syllable as /sak1-waa0/. However, we do think that syllabification is generally useful for pronunciation disambiguation. Unlike Meknavin and Kijsirikul. (2000), and Tesprasit et al. (2003), who jumped directly to the word level, we think that many problems can be solved at the syllable level. And it is at this level that we could deal with problems mentioned in the last section more efficiently.

Problems (a), (b), (c), and (d) could be disregarded if the correct sequence of syllables is determined. When we know the syllable boundary, we would know how to map each character to its corresponding sound (problem (a)); we could add the missing vowel to a syllable that has only consonants[7] (problem (b)); and we will not have problems (c) and (d) at all.

For problems (e), (f), (g), and (h), we think that it should not be solved at the character level, as did in previous research. They are specific characteristics of some syllables. In problem (e), pronunciation ambiguity of "ฑ" could not be predicted at the character level. Actually, there are only a few syllables that "ฑ" should be mapped to /d/. Problems (f) and (g) are found on some loan words from Pali and Sanskrit. They are not a productive process. So, they should be handled as exceptions of some syllables rather than handled by rules. In (h), these are words that their pronunciations do not comply with letter-to-sound rules. So, they should also be treated as exceptions of some syllables. As for problem (i), the ambiguity of pronunciation would not be easily resolved by considering only nearby characters. But it is easier to predict how these clusters should be pronounced by considering surrounding syllables. For example, if "ทรา" occurs after "จัน", as in "จันทรา"-/can0-thraa0/-{moon}, it will be pronounced /thraa0/; but if it occurs in between "กัน" and "ทร", as in "กันทรายทร"-/kan0-tha3+raa0-khoon0/-{mountain}, it will be pronounced /tha3+raa0/. Therefore, we believe that syllabification is a necessary step for grapheme-to-phoneme conversion.

In our model, unlike Kamya et al. (2000) who use rules for parsing syllables, syllabification is done on the basis of statistics. However, as stated before, we do not use statistical information at the character level as Chotimongkol and Black (2000), and Tarsaku et al. (2001) did, we use a trigram model of syllables to disambiguate syllable segmentations. The most probable syllable sequence is determined from the input characters. All possible pronunciations of each syllable are generated at this step too. After that, the right pronunciation of each syllable is chosen based on the result of word segmentation and the statistical information of pronunciation.

The model of transcription here can be viewed as a probabilistic model as below.

$$\begin{aligned}
&\arg\max_{w\&\,phw} P(w_1..w_n \;\&\; phw_1..phw_n \mid c_1..c_m) \\
&= \arg\max_{w\&\,phw} P(c_1..c_m \mid w_1..w_n \;\&\; phw_1..phw_n) \\
&\quad * P(w_1..w_n \;\&\; phw_1..phw_n) / P(c_1..c_m) \\
&= \arg\max_{w\&\,phw} P(w_1..w_n \;\&\; phw_1..phw_n) \\
&= \arg\max_{w\&\,phw} P(w_1..w_n) * P(phw_1..phw_n \mid w_1..w_n) \\
&\approx \arg\max_{w\&\,phw} P(w_1..w_n) * \prod_{i=1..n} P(phw_i \mid w_i)
\end{aligned}$$

[7] The missing vowel can be added correctly by considering consonants characters in the syllable.

$P(w_1...w_n$ & $phw_1...phw_n \mid c_1...c_m)$ is the probability that character strings $c_1..c_m$ will be word-segmented as $w_1..w_n$ and pronounced as $phw_1..phw_n$, such that phw_i is the pronunciation of w_i. The second line is equivalent by Bayes' rules. The third line is derived from the fact that $P(c_1..c_m)$ is a constant and $P(c_1..c_m \mid w_1..w_n$ & $phw_1..phw_n)$ is equal to one. Since we assume that the pronunciation of each word is not affected by other words, the probability of pronunciation of word sequence $w_1..w_n$ is estimated as the product of the pronunciation of each word. Thus, as seen in the last line, the model of Thai transcription is viewed as composing of two models: the language model and the pronunciation model.

3.1 Language Model

In this model, word sequences are produced from the sequence of input characters. Since syllabification is very useful for transcription, it will be included as a part of the model. We adopted Aroonmanakun's word segmentation model (2002) for this study. In his model, syllable segmentation is the first process. An input string is segmented into syllables by comparing to syllable patterns and syllable forms that are exceptions. For example, ıCRTาะ, ıXTอะ, ıCRTY are syllable patterns in which X, C, R, Y, T stands for a different group of characters. Pronunciations of syllables matched to syllable patterns are generated by rules. Syllables that do not conformed to the syllable patterns or their pronunciations are different from those generated by rules, such as "ธาตุ"-/thaat2/-{elements}, "อัฐ"-/?at1/-{ashes}, etc., are listed as exceptions. For example, "ธาตุ"-/thaat2/ is treated as an exception because it is uncommon to have the vowel form "ุ" under the final consonant. There are 220 syllable patterns and 1,935 exceptional syllables used in our system. The results after applying these syllable patterns and exceptions are usually ambiguous. For example, "ประโยคธรรมดา"-{simple sentence} could be syllable-segmented in three ways, namely "ประ-โยค-ธรา-มดา"-/pra1-jook1-than0-ma3+daa0/, "ประ-โยค-ธรรม-ดา"-/pra1-jook1-tham0+ma3-daa0/, "ประ-โยค-ธร-รม-ดา"-/pra1-jook1-thɔɔn0-rom0-daa0/. The most probable syllable segmentation is selected by the use of a trigram model. In this study, a training corpus of 638,277 syllables from newspapers is manually segmented. Witten-Bell discounting is used for smoothing (Chen and Goodman, 1998). Viterbi algorithm is used for determining the best segmentation. The selected sequence of syllables then will be grouped into words. This process is also non-deterministic. There could be many ways to group a given syllable sequence into a word sequence. We adopted Aroonmanakun's maximum collocation approach to select the best word sequence (Aroonmanakun 2002). Collocation strength of a word sequence is the sum of all words' collocation strengths in the sequence (Fw_i). Collocation strength of a word is the sum of collocation strengths between syllables in that word.

$$St = \sum_{i=1}^{n} F_{w_i} \quad F_{w_i} = \sum_{j=1}^{k-1} C_{s_j,s_{j+1}} \; such\ that\ w_i = s_1 s_2 ....s_k \; and\ s_j \; is\ a\ syllable$$

Collocation strength between syllables is the ratio of p(x,y) to q(x,y), where p(x,y) is the probability of finding syllables x and y together, and q(x,y) is the probability of finding any syllable in between x and y (x-ANY-y), or the probability for x and y to be separated by any syllable. The collocation between syllables x-y then is calculated as below:

$$\log \frac{p(x,y)}{q(x,y)} = \log \frac{p(x)p(y \mid x)}{q(x)q(y \mid x)} = \log \frac{p(y \mid x)}{q(y \mid x)} = \log \frac{Count(x,y)/Count(x)}{Count(x,Any,Y)/Count(x)} = \log \frac{Count(x,y)}{Count(x,Any,y)}$$

The output from this model will be the sequence of words, in which each syllable in a word is attached with all possible pronunciations.

3.2 Pronunciation Model

In this model, we assume that pronunciation of a syllable could be determined within the word. Surrounding words do not affect the pronunciation of the target word. In addition, it is assumed that the pronunciation of a syllable is affected only by the preceding and the following syllable forms. The pronunciation then can be estimated as follows:

$$P(phw_i \mid w_i) = P(phs_1..phs_k \mid s_1..s_k)$$
$$\approx \prod_{j=1..k} P(phs_j \mid s_{j-1}s_j s_{j+1})$$

$P(phw_i \mid w_i)$ is the probability that a given word, w_i, will be pronounced phw_i. If the word is composed of syllables $s_1..s_k$, the pronunciation phw_i will be $phs_1..phs_k$, where phs_j is the pronunciation of syllable s_j. $P(phs_j \mid s_{j-1}s_j s_{j+1})$ is the probability that syllable s_j will be pronounced phs_j when the preceding and following syllables are s_{j-1} and s_{j+1}. The probability is estimated from a corpus of word-pronunciation. It is a list of 28,620 words manually aligned between syllable and its transcription. These words are extracted from the Royal Institute dictioncary.

3.3 Romanizing the Transcription

At the final process, the transcription output from the system is adjusted to comply with the Royal Institute's guideline of romanization. Each phonetic sound is replaced with the corresponding roman characters, such as /ŋ/ is changed to "ng", /ɛ/ is changed to "ae", etc. Tone and vowel lengths are deleted. Hyphen is added between two syllables if the final character of the preceding syllable and the initial character of the following syllable may cause reading ambiguity. For example, "samang" is changed to "sam-ang"; "saat" is changed to "sa-at". Space is inserted as word separation, such as "prasop khwam samret" (meet – Nom. Marker – success). The first character of the word is capitalized when romanizing proper names, such as "Dekying Umbun Thongmi" (Girl – First Name – Last Name).

4 Experiments

The system was tested on two data sets: general texts and geographical names. A running text of 18,388 words extracted from a newspaper is used for the first test. The purpose of the first test is to test the accuracy of romanization for general texts. The second test set is the list of 990 Thai geographical names extracted from the Royal Institute's books. Since romanization is generally used for writing Thai geographical names, the purpose of the second test is to check the accuracy of the system for romanizing geographical names. The results are shown in Table 1.

	General texts	Names
Correct	18311 (99.58%)	935 (94.44%)
Incorrect	77 (0.42%)	55 (5.56%)
	18388 (100%)	990 (100%)

Table 1 : Result of romanization

From the results, we can see that the system is highly accurate for general texts, but not for geographical names. For 77 errors found in general texts, 8 errors are caused from abbreviations and 64 errors are caused from English transliterated words, such as "แชมป์"-(champ), "เทรนเนอร์"-(trainer). These words are not supposed to be romanized. They should be processed by a backward transliteration system to its original form in English. If we exclude these 72 instances of errors, the system can romanize Thai texts correctly at the level of 99.97% (18311/18316). Therefore, there are only 5 errors of romanization, caused from the following words: "ทุกจุ่น"-/ku3-krun1/-{glowing}, "ปอดบวม"-/pɔɔt1-buuam0/-{puemonia}, "พิเชษฐ"- /phi3-chet2/-{Name}, "พิพัฒน์พงศ์"-/phi3-phat3-phoŋ0/-{Name}, "อินทรัตน์"-/?in0+

tha3-rat3/-{Name}. Errors of the first two words are caused from errors in syllabification. The word "ฤกรุ่น"-{glowing} was incorrectly syllable-segmented as "ฤก-รุ่น", rather than "ฤ-กรุ่น". The word "ปอดบวม"-{puemonia} was wrongly segmented as "ปอ-ดบวม", rather than "ปอด-บวม".The last three errors are caused from Thai person names.

For 55 errors of geographical names, 4 errors are caused by incorrect syllable segmentation. These four names are segmented as "ขน-อม"-/khon4-?om0/, "บ้าน-เข-ว้า"-/baan2-khee4-waa4/, "ทุ่ง-เส-ลี่-ยม"-/thuŋ2-see4-lii2-jom0/, and "เข-วา-สิน-รินทร์"-/khee4-waa0-sin4-rin0/, while the correct ones should be "ขนอม"-/kha1+nɔɔm4/, "บ้าน-เขว้า"-/baan2-kha1+waw2/, "ทุ่ง-เสลี่ยม"-/thuŋ2-sa1+liiam1/, and "เขวา-สิน-รินทร์"-/khwaw4-sin4-rin0/ respectively. Eleven errors are caused by mispronunciation of seven syllables. For example, "สระ"-/sa1/ was mis-romanized as "sara" rather than "sa"; "แหง"-/hɛɛŋ4/ is mis-romanized as "ngae" rather than "haeng". The rest of errors (40), which are the majority of errors, are caused by incorrect word recognition. Failure to recognize word boundary results in the lack of linking syllables. For example, if the name "ราชเทวี"-/raat2+cha3-thee0-wii0/ is not recognized as one word, it will be incorrectly romanized as "rat thewi". But if it is recognized as one word, it should be romanized correctly with a linking syllable as "ratchathewi". In addition, since many Thai names are composed of loan words from Pali and Sanskrit, their pronunciations are quite different from those of general words, which are used for training in the pronunciation model. Adding a linking syllable is quite common when pronouncing these names. Thus, it is not surprising why the accuracy is dropped to 94.44% when romanizing these geographical names.

To verify whether errors were mainly caused from the unsuitable training corpus, 3,000 person names were added to the training data of the pronunciation model, and these 990 geographical names were retested again. In addition, to handle the missing linking syllable which is caused from wrong word recognition, the system was instructed to treat each input name as one word in this test.

	Names
Correct	961 (97.07%)
Incorrect	29 (2.93%)
	990 (100%)

Table 2 : Result of romanization

From Table 2, we can see that adding person names in the training data could increase the accuracy of romanization. However, some errors still remain. Errors from incorrect syllable segmentation (4 errors) and mispronunciation (11 errors) are not solved because the added names in the training corpus do not have any new information to solve these problems. Many errors caused from the lack of linking syllables are solved at this test because the system is instructed to view each geographical name as one word. For example, "ราชเทวี"-/raat2+cha3-thee0-wii0/ is correctly romanized with a linking syllable as "ratchathewi". However, by treating each name as one word, the system fails to romanize some names because some geographical names may compose of more than one word. For example, "พนัสนิคม"-/pha3+nat3-ni3-khom0/ should be romanized without the linking syllable, "phanat nikhom", because this name is composed of two words "พนัส"-/pha3+nat3/-{forest} and "นิคม"-/ni3-khom0/-{settlement}. But the system mis-romanized it with a linking syllable as "phanatsanikhom".

5 Discussion

The results show that the accuracy can be increased if appropriate training data is added. Since the current training data are mostly general texts, more training data on proper names should be added to the system. In addition, from the experiments, our system cannot romanize homographs correctly. For example, "สระ" can be romanized as either "sa" or "sara" depending on its meaning. Since we assume that pronunciation of syllables could be determined within the word, the system will not know how to disambiguate the pronunciations in these cases. Luckily homographs with different sounds are rare in

Thai. (There are only 8 words.) Most of them are hardly used in general texts. Only two words, "พระ"-{pond ; vowel} and "แพน"-{Name of plant ; be jealous}, need to be solved in the future development.

The results also show that the recognition of word boundary is very important for romanization. Incorrect word segmentation can cause an error in romanization. Thus, performance of romanization system relies heavily on the word segmentation algorithm. Unfortunately, none of Thai word segmentation algorithms yields perfect results. In addition, to romanize any texts according to the guideline of the Royal Institute, the system must be able to identify not only new words, but also proper names and their types. If the system is romanizing a person name, it should analyze that name as one word rather than composition of words. But if it is romanizing a geographical name, it should break down that name into words and insert a space for word separation. For example, if "แก่งหางแมว"-/kɛɛŋ1-haaŋ4-mɛɛw0/-{rapids-tail-cat} is a person name, it should be romanized as "Kaenghangmaeo". But if it is a geographical name, it should be romanized as "Kaeng Hang Maeo". Therefore, identification of new words and proper names should also be implemented in the next development. But at present, to make the system suitable as a public tool for romanization, the system must allow users to specify whether the input text should be treated as a single word or running texts, and allow users to add new words in a user dictionary. This would lessen the problems.

The results also indicate that the statistical model works very well for the transcription task. Our system which is based on a simple n-gram model trained with a moderate-size corpus can perform quite well. In our view, the accuracy depends on the system design rather than the statistical method. Our system is designed to work at the syllable level, which is more suitable to the Thai transcription problem than systems that work mainly at the character level. Therefore, any NLP systems should be carefully designed to suit the analysis of the linguistic problems. Statistical models are not by itself a magic box which can solve any linguistic problems. Linguistic analysis of the problems should be done first to understand the nature the problems. Moreover, we can see that error analysis is necessary for improving the system's performance. By locating where the problems are, we can prepare the training data that are suitable for the task. We can also see what could be improved and what would be the limitation of the current design.

Acknowledgements

We would like to thank the Thailand Research Fund and the Commission on Higher Education for funding this research. A Thai dictionary used in this system is mainly extracted from the Royal Institute dictionary, which is made available by the Thailand's Information Research and Development Division, National Electronics and Computer Technology Center.

References

ALA-LC 1997. *Romanization tables : transliteration schemes for non-roman scripts 1997*, approved by the Library of Congress and the American Libraty Association, Randall K. Barry (editor). Washington, D.C.: Library of Congress.

Al-Onaizan, Y. and K. Knight. 2002. Machine Transliteration of Names in Arabic Text. In *Proceedings of ACL Workshop on Computational Approaches to Semitic Languages*, pages 34-46, Philadelphia, USA.

Aroonmanakun, W. 2002. Collocation and Thai Word Segmentation. In *Proceedings of the Fifth Symposium on Natural Language Processing & The Fifth Oriental COCOSDA Workshop*, pages 68-75, Pathumthani: Sirindhorn International Institute of Technology.

Bosch, A. van den and W. Daelemans. 1993. Data-oriented methods for grapheme-to-phoneme conversion. In *Proceedings of the Sixth Conference of the European Chapter of the Association for Computational Linguistics*, pages 45-53, Utrecht, Netherland.

Chen, Stanley F., and Joshua Goodman. 1998. *An Empirical Study of Smoothing techniques for Language Modeling*. TR-10-98. Harvard University.

Chotimongkol, Ananlada and Alan W Black. Statistically trained orthographic to sound Models for Thai, In *Proceedings of ICSLP 2000*, Beijing, China October 2000.

Frankfurt, O. 1906. Some Suggestions for Romanizing Siamese. In *Journal of the Siam Society*, Vol.3, Part 2, 52-61.

Frankfurt, O., P. Petithuguenin, and J. Crosby. 1931. Proposed System for the Transliteration of Siamese Words into Roman Characters. In *Journal of the Siam Society*, Vol.10, Part 4, 1-22.

Griswold, A. B. 1960. Afterthoughts on the Romanization of Siamese. In *Journal of the Siam Society*, Vol.48, Part 1, 29-66.

ISO/FDIS 11940. 1998. *Information and documentation --- transliteration of Thai*, (ISO 11940 : 1998)

Khamya, A., L. Narupiyakul, and B. Sirinaovakul, 2000. SATTS : Syllable Analysis for Text-To-Speech System, In *The 4th Symposium on Natural Language Processing (SNLP 2000)*, pages 336-340, May 10-12, Chiang Mai Plaza Hotel, Chiang Mai.

Knight, Kevin and Jonathan Graehl. 1997. Machine Transliteration. In *Proceedings of 35th Annual Meeting of the Association for Computational Linguistics and the 8th Conference of the European Chapter of the Association for Computational Linguistics*, pages 128-135, Madrid, Spain.

Londe, David L., U. Warotamasikkhadit, and N. Kanchanawan. 1971. *TRACTS : Thai-Roman Computerized Transliteration System.* Research Report sponsored by the Advanced Research Projects Agency for the Thai-US Military Research and Development Center.

Luksaneeyanawin, Sudaporn. 1989. A Thai Text to Speech System. In *Proceeding of the Conference of the Regional Workshops on Computer Processing of Asian Languages*, pages 305-315, Asian Institute of Technology.

Meknavin, Surapant and Boonserm Kijsirikul. 2000. Thai Grapheme-to-Phoneme Conversion. In Burnham, Denis, et.al., editors, *Interdisciplinary Approaches to Language Processing: The International Conference on Human and Machine Processing of Language and Speech*. NECTEC: Bangkok.

Petithuguenin, P. 1912. Method for Romanizing Siamese. In *Journal of the Siam Society*, Vol.9, Part 3, 1-12.

Royal Institute. 1999. *Principles of Romanization for Thai Script by Transcription Method.*

Tarsaku, Pongthai, Virach Sornlertlamvanich, and Rachod Thongpresirt. 2001. Thai Grapheme-to-Phoneme Using Probabilistic GLR Parser. In *Proceedings of Eurospeech 2001*, Aalborg, Denmark, Sept 2001.

Tesprasit, Virongrong, Paisarn Charoenpornsawat, and Virach Sornlertlamvanich. 2003. A Context-Sensitive Homograph Disambiguation in Thai Text-to-Speech Synthesis. In *Human Language Technology Conference 2003* (HLT-NAACL 2003), Edmonton, Canada, May 2003

Thiengburanathum, Wit, 1977. *Thai-English Dictionary.* Ruamsan Co.,Ltd., Bangkok: 2539.

United Nation. 2002. *Report on the current status of the United Nation romanization systems for geographical names.*

Vajiravudh, King. 1912. The Romanization of Siamese Words. In *Journal of the Siam Society*, Vol.9, Part 4, 1-10.

Vajiravudh, King. 1931. Notes on the Proposed System for the Transliteration of Siamese Words into Roman Characters. In *Journal of the Siam Society*, Vol.10, Part 4, 24-33.

Table of the Royal Institute's Thai Romanization

Consonant form	Romanized character		Vowel form	Romanized character
	Initial con.	Final con.		
ก	k	k	อะ, ั (reduced form of อะ) , รร (with final consonant), อา	a
ข ฃ ค ฅ ฆ	kh	k	รร (without final consonant)	an
ง	ng	ng	อำ	am
จ ฉ ช ฌ	ch	t	อิ, อี	i
ซ ทร (pronounced ซ) ศ ษ ส	s	t	อึ, อื	ue
ญ	y	n	อุ, อู	u
ฎ ฑ (pronounced ด) ด	d	t	เอะ, เ็ (reduced form of เอะ), เอ	e
ฏ ต	t	t	แอะ, แอ	ae
ฐ ฑ ฒ ถ ท ธ	th	t	โอะ, -(reduced form of โอะ), โอ , เอาะ, ออ	o
ณ น	n	n	เออะ, เ็ (reduced form of เออะ) , เออ	oe
บ	b	p	เอียะ, เอีย	ia
ป	p	p	เอือะ, เอือ	uea
ผ พ ภ	ph	p	อัวะ, อัว, -ว- (reduced form of อัว)	ua
ฝ ฟ	f	p	ไอ, ไอ, อัย, ไอย, อาย	ai
ม	m	m	เอา, อาว	ao
ย	y	–	อย	ui
ร	r	n	ไอย, ออย	oi
ล ฬ	l	n	เอย	oei
ว	w	–	เอือย	ueai
ห ฮ	h	–	อวย	uai
			อิว	io
			เอิว, เอว	eo
			แอิว, แอว	aeo
			เอียว	iao
			ฤ (pronounced รึ), ฤๅ	rue
			ฤ (pronounced ริ)	ri
			ฤ (pronounced เรอ)	roe
			ฦ, ฦๅ	lue

A Chunk-based n-gram English to Thai Transliteration

Abstract

In this study, a chunk-based n-gram model is proposed for English to Thai transliteration. The model is compared with three models: table lookup model, decision tree model, statistical model. The chunk-based n-gram model achieves 68% word accuracy, which is better than the accuracy of other models. Performances of all models are increased when an English grapheme to phoneme is included in the system. However, 68-84% accuracy is not sufficient to be used by the public. An investigation of the problems and some suggestions are provided. We believe that the low accuracy of the system is caused by the poor performance of the English grapheme to phoneme module and the inconsistency of pronunication in the training data.

1 Introduction

English to Thai transliteration is a way to write English words in Thai alphabets. While English has 26 characters for consonant and vowel sounds, Thai has 44 characters for 21 consonant sounds, and 19 characters (including 3 consonant characters) for 24 vowel sounds (including 6 diphthongs), and 4 characters for tone markers. When transliterating English words into Thai words, it is usual to have different Thai written forms. For example, the word "internet" can be found written as อินเตอร์เน็ต, อินเทอร์เน็ต, อินเตอร์เนท, or อินเตอร์เนต. To minimize the varieties of transliterated words, the Thai Royal Institute issued the regulations of English-Thai transliteration in 1982. Nevertheless, many people tend to transliterate English words on their own rather than adhering to the regulation. It would be more convenient for people to write transliterated words if an automatic English to Thai transliteration program that conforms to the Royal Institute's guideline is available. In this study, we aim to develop such a system. A corpus of transliterated words is created by collecting English and Thai word pairs from books published by the Royal Institute. The number of 8,181 word pairs are used in this study. In each word pair, Thai characters are aligned with their English correspondent characters. Alignments between English and Thai characters are first assigned by a program and then manually corrected. It is possible that more than one character in English or Thai are aligned, e.g. 'th'-'ท', 'ıa'-'ีย'. Examples of aligned characters between word pairs are shown below. These data will be used for training and testing.

l/ i/ th/ o/ s/ o/ l/ s/	ล/ ิ/ ท/ โ/ ช/ อ/ ล/ ์/
l/ i/ th/ ua/ n/ ia/	ล/ ิ/ ท/ ว/ น/ เ/ ีย/
l/ i/ v/ e/ r/ p/ oo/ l/	ล/ ิ/ ว/ เ/ ร/ ์/ พ/ ู/ ล/
l/ i/ v/ i/ ng/ s/ t/ o/ n/ e/	ล/ ิ/ ฟ/ ว/ ิ/ ง/ ส/ ต/ โ/ น/ #/
l/ i/ v/ i/ u/ s/	ล/ ิ/ ว/ ิ/ อ/ ์/ ส/

This paper first reviews previous models of transliteration systems. Table lookup, decision tree, and statistical models are briefly discussed. Then, a new approach of chunk-based n-gram model is explained in section 3. The results when using each model are reported and compared in section 4. Since knowing English pronunciation is usually useful for transliteration, all the models are tested again by including a module of English grapheme to phoneme. The new results are reported in section 5. Though the chunk-based n-gram model performs better than other models, the accuracy is not high enough to be used as a tool for the public. At the end, we will review and discuss the problems.

2 Previous research

Since transliteration is basically a process of transforming one writing system into another writing system, approaches used in any transliteration systems as well as grapheme to

phoneme systems are relevant. In this study, three different approaches are reviewed and implemented, namely table lookup, decision trees, and noisy channel model.

2.1 Table Lookup Model

The model is based on Bosch and Daelemans' grapheme-to-phoneme conversion model (1993). It is used for transcribe English, French, and Dutch. Conversion of characters to phonemes is done by applying conversion rules, which are extracted from a training corpus. But in this study, the conversion rules are used for converting characters from one language to another. Rules are store in a table lookup as a mapping from source language characters to target language characters, with left and right contexts of the source language as rule conditions. By using a training corpus composing of word pairs aligned between characters of the two languages, if the target language character can be uniquely determined from the source language character within its minimal context, the conversion rule will be stored in table lookup. But when the same context does not uniquely determine the target language character, conversion rules are done by default mapping by selecting the most occurring target character in that context. In this study, table lookup of various context sizes are implemented: 0-0, 0-1, 1-1, 1-2, 2-2, 2-3, 3-3, 3-4, 4-4, 4-5, and 5-5. (The two digits indicate the number of characters on the left and right contexts) Default mapping of 0-0, 1-1, and 2-2 contexts are used when table lookup is not applicable.

2.2 Decision Tree Model

Decision tree model converts symbols from one language to another by applying rules that are in the form of decision tree. Kang and Choi (2000) use this model for Korean-English transliteration system. Decision tree is created by applying a well-known machine learning technique, ID3. This method is often applied to many NLP systems, such as Thai grapheme to phoneme (Chotimongkol and Black 2000), word sense disambiguation (Pedersen 2004). In this study, Lenzo's (1998) decision tree model for English grapheme to phoneme is modified to create decision trees for English to Thai transliteration. The maximum depth of trees is set to 7. The scope of contexts are set to 3 characters for both left and right contexts.

2.3 Statistical Model

The third model is a statistical model, which is often used in transliteration research, such as Japanese-English back-transliteration (Knight and Graehl 1997), English-Arabic transliteration (Glover and Knight 1998), English-Korean transliteration (Kang and Kim 2000), English/Japanese transliteration (Fujii and Ishikawa 1999, 2001), English-Korean transliteration (Jung et al. 2000), etc. Transliteration problem is viewed as a probabilistic model. In this study, English-Thai transliteration can be viewed in a similar way as follows:

$$\arg\max_{T_w} P(T_w \mid E_w) = \arg\max_{T_w} \frac{P(T_w) * P(E_w \mid T_w)}{P(E_w)}$$

$$= \arg\max_{T_w} P(T_w) * P(E_w \mid T_w)$$

$$P(T_w) = P(Tc_1, Tc_2, \ldots, Tc_n) \approx \prod_{i=1,n} P(Tc_i \mid Tc_{i-2} Tc_{i-1})$$

$$P(E_w \mid T_w) \approx \prod_{i=1,n} P(Ec_i \mid Tc_i)$$

Transliteration from English to Thai is composed of two sub-models: P(T) and P(E|T). P(T) can be estimated by a trigram model, while P(E|T) is estimated from alignments between English and Thai characters in the training corpus.

In addition, Haizhou et al. (2004) joint source-channel model, which is used for English-Chinese transliteration, will be implemented as another test model in this study. Unlike other models which capture how source words can be mapped to target words, this model uses both source and target words simultaneously. The model can be formulated as follows:

$$\arg\max_T P(T \mid E) = \arg\max_T \frac{P(T,E)}{P(E)} = \arg\max_T P(T,E)$$

$$= \arg\max_T P(t_1 t_2 \ldots t_n, e_1 e_2 \ldots e_n)$$

$$= \arg\max_T P(< t_1, e_1 >, < t_2, e_2 >, \ldots < t_n, e_n >)$$

$$\approx \arg\max_T \prod_{i=1}^{n} P(< t_i, e_i > \mid < t_{i-2}, e_{i-2} >< t_{i-1}, e_{i-1} >)$$

Transliteration is viewed as a probabilistic model of transliteration pairs between English and Thai characters, which then can be estimated by a trigram model.

3 Chunk-based n-gram Model

The model proposed in this study is a chunk-based n-gram model. It is based on Kang and Kim's (2000) view of phoneme chunks and Haizhou et al.'s (2004) joint source-channel model. In this chunk-based model, alignment between English and Thai characters can have various lengths. For example, for the word pair "locarno" โลคาร์โน, beside the normal alignments l-ล, o-โ, c-ค, a-า, r-ร์, n-น, o-โ, alignments of larger units, i.e. lo-ลโ, oc-โค, ca-คา, ar-าร์, rn-ร์น, no-นโ, loc-ลโค, oca-โคา, car-คาร์, arn-าร์น, rno-ร์นโ, ..., and locarno-ลโคาร์นโ are also generated. Like other statistical models, transliteration here is viewed as a probabilistic model of transliteration pairs between Thai and English. But in this model, the units of transliteration pair can be a chunk of characters Probability of a sequence of transliteration pairs is estimated by a trigram model in this study. The sequence with the highest probability will be selected as the solution.

$$\arg\max_T P(T \mid E) = \arg\max_T \frac{P(T,E)}{P(E)} = \arg\max_T P(T,E)$$

$$= \arg\max_T P(t_1 t_2 \ldots t_m, e_1 e_2 \ldots e_n)$$

$$= \arg\max_T P(< t_{1..a}, e_{1..a} >, < t_{a+1..b}, e_{a+1..b} >, \ldots < t_{m+1..n}, e_{m+1..n} >)$$

$$= \arg\max P(< ct_1, ce_1 >, < ct_2, ce_2 >, \ldots < ct_n, ce_n >)$$

$$\approx \prod_{i=1}^{n} P(< ct_i, ce_i > | < ct_{i-2}, ce_{i-2} >< ct_{i-1}, ce_{i-1} >)$$

(ci, and ei, are a chunck of Thai and English characters)

For example, when transliterating 'unitarian', there could be many possible sequences of transliteration pairs as follows:

<unita,ยูนิทเ.>,<r,ร>,<ian,เ็ยน>

<unita,ยูนิทเ.>,<r,ร>,<i,ไ.><an,อัน>

<unitar,ยูนิทเ.ร>,<ian,อาน>

<uni,ยูนไ.>,<t,ท>,<a,า>,<rian,รเ็ยน>

<unit,ยูนิท>,<arian,าเรเ็ยน>

....

Probability of each sequence is calculated based on trigram statistics of all transliteration pairs in the training data. Using this model, it is likely that a sequence with fewer chunks will have higher probability than a sequence with longer chunks and chunks with high occurrences will have higher probability than chunks with low occurrences.

4 Experiments

The corpus of 8,181 English-Thai pairs is splitted into five data sets. Four of them are used as the training data, while the other one is used as the test data. Each system will be tested on each data set. Then, the results of five tests will be averaged as the performance of each system on unseen data. Each system is also tested for seen data (All data sets are used as training and testing). Performance is measured in two ways: word accuracy (W.A.) and character accuracy (C.A.) (Kang and Kim 2000). W.A. is counted from the exact match of the generated words and the correct word. C.A. is calculated on the basis of edit distance between the two words. C.A. = (L – (i + d +s)) / L where L is the length of a word, and i, d, s is the number of insertion, deletion, and substitution that are needed to change the result to match the target word.

The results are shown in Table 1. It can be seen that the chunk-based n-gram performs better than other systems for both unseen and seen data. For unseen words, the accuracy at the word level is 67.4%, and the accuracy at the chracter level is 88.9% For seen words, word accuracy of the chunk-based n-gram is 99.8% and chracter accuracy is 99.9%

	Unseen		Seen	
	W.A.	C.A.	W.A.	C.A.
TB	60.6%	85.8%	97.2%	99.2%
DT	61.7%	86.8%	95.7%	99.0%
N-gram	37.6%	77.1%	48.2%	82.1%
Joint	50.1%	84.4%	67.2%	90.0%
Chunk	67.4%	88.9%	99.8%	99.9%

Table 1: Result of English-to-Thai transliteration

5 Adding E2P module

Knowing how the word is pronounced is usually useful for transliteration from English to Thai. In fact, to transliterate an English vowel correctly, it is necessary to know how it is pronounced. For example, vowel form 'i' can be transliterated to three Thai vowel forms, ิ ี ไ depending on the pronunciation of that vowel. Therefore, the systems are tested again by adding a module of English grapheme to phoneme (E2P) as a part of the system. But only three models, TB, DT, and Chunk-based, are tested at this time. The systems are implemented

by considering both English characters and phonemes in this new test. E2P module is created by applying table lookup, which are generated from a CMU pronunciation dictionary. The accuracy of the E2P is measured at 56.7% for W.A. and 90.8% C.A. The new result of the systems when including E2P is shown in Table 2.

	Unseen		Seen	
	W.A	C.A	W.A	C.A
TB	65.3%	88.5%	99.7%	99.5%
DT	63.0%	87.4%	97.4%	99.4%
Chunk	68.1%	88.7%	99.8%	99.9%

Table 2 : Result when E2P is included

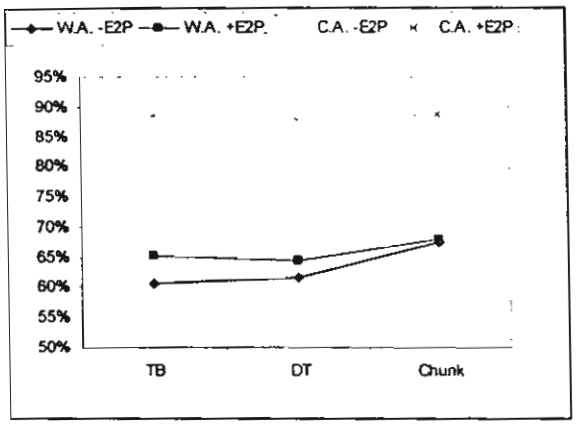


Figure 1: Results with or without E2P

It can be seen that all systems perform a bit better when English G2P is included. And the chunk-based system is still the best model in this study.

6 Discussions

Although the chunk-based n-gram model performs better than other models, the accuracy of the system is still not high enough for using as a transliteration tool for the public. To further improve the performance, error analysis is needed to understand why the generated words do not match the correct words.

Two factors are likely the causes of low accuracy: accuracy of E2P and varieties of pronunciation. The accuracy of E2P module developed in this study is not really high. It yields only 56.7% for word accuracy and 90.8% for phoneme accuracy. Creating a good E2P module is difficult. Black et al. (1998) also reported the accuracy of their E2P system at 57.80% for word accuracy and 91.99 for phoneme accuracy when using CMU pronunciation dictionary as training and testing data, while the accuracy when using Oxford Advanced Learners Dictionary of Contemporary English is higher (74.56%). Black et al. explained that the difference lies on the fact that CMU dictionary include a lot more proper names. According to Llitjós (2001), systems that produce high accurate results are those that are dictionary-based. Transliteration rules are used only when the input word is not listed in the dictionary. But in English to Thai transliteration, many inputs are proper names. It is unlikely to have all names listed in the dictionary. Therefore, it is still necessary to create a good E2P module that is not dictionary-based.

Varieties of pronunciation could also be another cause of systems' low accuracy. Since transliteration is partly based on the pronunciation of English words and the same word can be pronounced differently, i.e. British or American pronunciations, the transliterated words then could be written differently. For example, the word "Leonard" is found transliterated in the Royal Institute's books as เลอนาร์ด, เลนเนิร์ด, and เลียวนาร์ด. Although only one form that we think is most conformed to the guideline is stored in the corpus in this case, it does not exclude the possibility of different accents entailed in different words. This inconsistency of the data results in inefficiency of transliteration rules extracted from the corpus.

In addition, when the generated words do not exactly match the correct words, it is possible that the generated word is another form of acceptable transliterations. For example, the word "ballast" is transliterated by the chunk-based system as บัลลาสต์ while the correct word is แบลลาสต์. The difference of vowel form in this case is resulted from different accents of pronunciation. In this example, the generated word is considered an acceptable result. Therefore, the results are manually checked whether they are acceptable transliteration. Using this acceptable criterion, it is found that the accuracy of the chunk-based model gains up to 84%.

7 Conclusion

Although the chunk-based model performs better other systems, 68-84% W.A. is not satisfying result. To be released as an English to Thai transliteration tool for the public, the program should have high accuracy up to 98%. The system has to be improved by employing a good E2P. And it might be necessary to manually clean up the training data to make English pronunciation in the transliterated words consistent with one particular accent. Beside, the chunk-based model uses more resources than other models. It is needed to be improved in terms of processing speed.

Acknowledgment

This research is supported by a grant from the Thailand Research Fund and the Commission on Higher Education.

References

Black, A. W., Lenzo, K. and Pagel, V. 1998. Issues in Building General Letter to Sound Rules. *3rd ESCA Speech Synthesis Workshop*, pp. 77-80, Jenolan Caves, Australia.

Bosch, A. van den and W. Daelemans. 1993. Data-oriented methods for grapheme-to-phoneme conversion. In *Proceedings of the Sixth Conference of the European Chapter of the Association for Computational Linguistics*, pages 45-53, Utrecht, Netherland.

Chotimongkol, Ananlada and Alan W Black. 2000. Statistically trained orthographic to sound Models for Thai, In *Proceedings of ICSLP 2000*, Beijing, China October 2000.

Fujii, Atsushi and Tetsuya Ishikawa. 1999. Cross-Language Information Retrieval for Technical Documents. In *Proceedings of the Joint ACL SIGDAT Conference on Empirical Methods in Natural Language Processing and Very Large Corpora*, pages 29-37,

Fujii, Atsushi and Tetsuya Ishikawa. 2001. Japanese/English Cross-Language Information Retrieval: Exploration of Query Translation and Transliteration. In *Computers and the Humanities, 35(4)*: 389-420.

Glover, Bonnie and Kevin Knight. 1998. Translating names and technical terms in arabic text. In *Proceedings of the COLING/ACL Workshop on Computational Approaches to Semitic Languages.* pages 34-41, Montreal, Quebec, Canada.

Haizhou, Li, Zhang Min, and Su Jian. 2004. A Joint Source-Channel Model for Machine Transliteration. In *Proceedings of the 42nd Annual Meeting of the Association for Computational Linguistics*, pages 159-166, Barcelona, Spain.

Jung, SungYoung, SungLim Hong, and Eunok Paek. 2000. An English to Korean Transliteration Model of Extended Markov Window. In *Proceedings of the 18th International Conference on Computational Linguistics*, pages 383-389, Saarbrücken, Germany.

Kang, B. J. and K. S. Choi (2000) Automatic transliteration and back-transliteration by decision tree learning. In *Proceedings of the 2nd International Conference on Language Resources and Evaluation*, Athnes, Greece.

Kang, In-Ho and GilChang Kim. 2000. English-to-Korean Transliteration using Multiple Unbounded Overlapping Phoneme Chunks. In *Proceedings of the 18th International Conference on Computational Linguistics*, pages 418-424, Saarbrücken, Germany.

Knight, Kevin and Jonathan Graehl. 1997. Machine Transliteration. In *Proceedings of 35th Annual Meeting of the Association for Computational Linguistics and the 8th Conference of the European Chapter of the Association for Computational Linguistics*, pages 128-135, Madrid, Spain.

Lenzo, Kevin A. (1998) "s/ ($text) / speech $1 / eg;" In *The Perl Journal*, Issue 12, pp.26-29.

Llitjós, Ariadna Font. 2001. *Improving Pronunciation Accuracy of Proper Names with Language Origin Classes.* Master Thesis. Carnegie Mellon University.

Pedersen, Ted. 2001. Lexical Semantic Ambiguity Resolution with Bigram-Based Decision Trees. In *Proceedings of CICLing 2001*, Mexico City, Mexico, Febuary 18-24, 2001, 157-168.

Thai Transcription (Version 3.0)

Submit
Reset

Process
IPA
Roman

Type in a Thai text and select the process and mode of transcription you want.

- IPA : is the transcription that includes consonant/vowel sounds and tone markers in Thai, using International Phonetic Alphabet. Select this process if your know how to read IPA. This process gives you more precise pronunciation than the Romanization.
- Roman : is the transcription of Thai sounds without tone markers, using the romanization guideline proposed by The Royal Institute (1982). This helps you transcribe Thai words by using English alphabets. The guideline is designed as a transcription of street and city names. For example, **เชียงใหม่** is transcribed as chiang-mai. (- is a syllable separator)
- "Automatic" mode picks the best transcription while "all" mode lists all possible transcriptions.
- To download "Thai Romanization" program for Windows, click <here>.

Mode
all
automatic

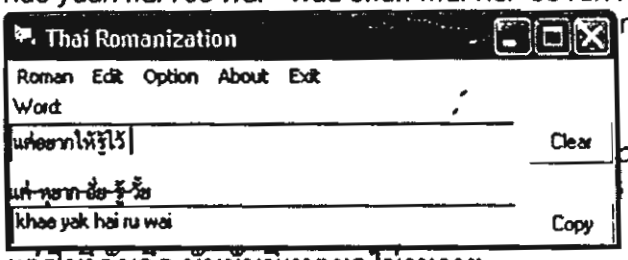
This project is supported by a grant from the Thailand Research Fund and the Commission on Higher Education (2003-4).
โครงการนี้ได้รับทุนสนับสนุนจาก ทุนพัฒนาศักยภาพในการทำงานวิจัยของอาจารย์รุ่นใหม่ สกว. ร่วมกับ สำนักงานคณะกรรมการการอุดมศึกษา ปี 2546-7

ThaiPhrasebook.com *Thai Language Books & Resources*

TITLE: Thai Romanization

แค่อยากให้รู้ไว้ ว่าฉันไม่เคยรักใครเท่าเธอ

kae yaak hai roo wai waa chun mai ker-ee ruk krai te



nyone

o roo

you.

แต่สิ่งที่ฉันคิด นั่นมันมีเหตุผลใช่ไหมจ๊ะ

AUTHOR: Chulalongkorn University

Homepage: www.arts.chula.ac.th

DESCRIPTION: A useful program for people who cannot read Thai yet. The program transcribes Thai words into readable Roman characters. This uses the Royal Institute's guideline for the "transliteration of Thai characters into Latin characters". Although the system is useable, it is not the same as web sites such as ethaimusic.com. For example, this web site would say "kae yaak hai roo wai" while this program will transcribe it as "khae yak hai ru wai".

DOWNLOAD: http://www.arts.chula.ac.th/~ling/tts/download.html

Have you bought this book? Please write a review.

SOFTWARE DESCRIPTION

Thai Romanization : โปรแกรมสำหรับแปลงข้อความภาษาไทยเป็นตัวเขียนในภาษาอังกฤษ ตามหลักเกณฑ์การถอดอักษรไทยเป็นโรมัน ของ ราชบัณฑิตยสถาน มีทั้งเวอร์ชั่นที่เป็น online และที่เป็น stand alone

Note : โปรแกรมนี้ ทางผู้พัฒนา โปรแกรม (Program Developer) เขาได้แจกให้ทุกท่านได้นำไปใช้ กับ ฟรี FREE !! นะครับผม โดยท่าน ไม่ต้องเสียค่าใช้จ่ายใดๆ ทั้งสิ้น ครับผม โดยท่านสามารถที่จะติดต่อกับทาง ผู้พัฒนาโปรแกรมนี้ได้ทาง E-Mail (ภาษาไทย) นะครับผม ...

English Description :

- **1.** IPA : is the transcription that includes consonant/vowel sounds and tone markers in Thai, using International Phonetic Alphabet. Select this process if your know how to read IPA. This process gives you more precise pronunciation than the Romanization.
- **2.** Roman : is the transcription of Thai sounds without tone markers, using the romanization guideline proposed by The Royal Institute (1982). This helps you transcribe Thai words by using English alphabets. The guideline is designed as a transcription of street and city names. For example, เชียงใหม่ is transcribed as chiang-mai. (- is a syllable separator)
- **3.** "Automatic" mode picks the best transcription while "all" mode lists all possible transcriptions.

INFORMATION

Thai Romanization

From : วิโรจน์ อรุณมานะกุล
Version : 1.04
Update : 3 พฤษภาคม ค.ศ. 2004
File Size : 3.6 MB.
OS : Windows 95 / 98 / ME / XP
License : Freeware
Downloads : 5,468

User Rated : 100% / 0% (11 Votes)
ให้คะแนนซอฟต์แวร์นี้
ดี ไม่ดี ส่ง

FEATURES

- Include Un-Installer

DOWNLOAD SOFTWARE

คุณสามารถ "ดาวน์โหลด" โปรแกรมนี้ได้ โดยการคลิกลิงค์ด้านล่างนี้

www.arts.chula.ac.th

LATEST FREEWARE ADDED

BUG REPORT AND DISCUSSION WEBBOARD

สำหรับใครที่มีปัญหาต่างๆ เกี่ยวกับตัว โปรแกรม / เกมส์ ไม่ว่าจะเป็นวิธีใช้หรือต้องการแจ้งข้อผิดพลาด (BUG) ของโปรแกรมนี้ คุณสามารถเข้าไปได้ที่นี่เลยครับ .

ENTER WEBBOARD